プロキシ再暗号化と検索タグを用いた範囲クエリ可能な秘匿情報の耐結託共有手法

平田　拓三† 　宮沢　駿輔† 　Hieu Hanh Le† 　横田　治夫†

† 東京工業大学　〒 152-8550 東京都目黒区大岡山 2-12-1

E-mail: †{hirata,miyazawa, hanhlh}@de.cs.titech.ac.jp, †yokota@cs.titech.ac.jp

あらまし　今日，プライバシーに関わる機微なデータがデータベースに保存される機会が増えており，データを保護するために適切なアクセス制御や暗号化が必要である．従来の暗号化データの共有方法では，共有グループからユーザの権限を削除する際に暗号化をし直さなければならなかった．プロキシによる再暗号化を用いた先行研究では，これを解決したほか，個人間の関係によるアクセス制御を導入した．しかしこの先行研究には，ユーザと信頼できないプロキシまたはデータベース管理者の結託によってデータの機密性が失われる，プロキシまたはデータベース管理者と結託しているユーザの権限削除ができない，範囲クエリに対応できない，個人間の関係を利用したアクセス制御が結託に対してセキュアでないという課題がある．本研究ではこれらの課題を解決する手法を提案し，セキュリティに関する考察を行う．

キーワード　データ共有，リボケーション，プロキシ再暗号化，範囲クエリ，耐結託

1. はじめに

今日，プライバシーに関わる機微な情報がデータベースに蓄積される機会が増えている．例えば，大規模な災害発生時において情報共有や被災者救援活動を支援するために情報技術を活用したシステムを利用することが注目されており [1] [2]，被災者の個人情報など，機密性のある情報が保存される可能性が高い．

そのようなシステムでは，データを保護するために適切なアクセス制御を行うことが必要である．さらに，データベース管理者が信頼できない場合や漏洩に備えてデータを暗号化して保存することが有効である．

従来の暗号化データの共有方法では，ユーザをリボケーション（すなわち共有グループからユーザの権限を削除）する際にデータを暗号化しなおした上，復号するための新しい鍵を再配布する必要があり非効率であった．児玉らは，この問題点をプロキシ再暗号化可能な暗号を用いることで解決した [4]．さらに，従来行われていたユーザのアクセスレベルによる制御に加えて，個人間の関係を利用したアクセス制御を導入した．

しかしこの研究には，ユーザとプロキシまたはデータベース管理者（DBA）の結託によってデータの機密性が失われる，プロキシまたは DBA と結託しているユーザのリボケーションができない，範囲クエリに対応できない，個人間の関係を利用したアクセス制御が結託に対してセキュアでない，といった課題がある．

そこで本研究では，これらの課題を解決することを目的として暗号化方法と，一致比較，大小比較が可能な検索タグの作成方法，個人間の関係を利用したセキュアなアクセス制御方法を提案する．

本論文では，2. 節で前提となる知識を述べる．3. 節で本研究の基となる先行研究として，プロキシ再暗号化を利用したアクセス制御手法を紹介し，4. 節でその残された課題を示す．5. 節で本研究の目的と提案手法を述べ，6. 節で提案手法をセキュリティの観点から考察する．最後に 7. 節でまとめと今後の課題を述べる．

2. 前提知識

本節では，提案手法を理解する際に前提とする知識を述べる．

2.1 プロキシ再暗号化

プロキシ再暗号化は，暗号文の変換によって復号を委譲する方法である [3]．従来の暗号化方式では，A 宛て，つまり A の公開鍵で暗号化された暗号文を B 宛ての暗号文に変換するためには，A は自分の秘密鍵 sk_A をプロキシに渡し，プロキシは sk_A により一度暗号文を復号してから B の公開鍵で暗号化する必要がある．しかしプロキシを信頼していない場合，平文へのアクセスを許可することは望ましくない．

プロキシ再暗号化可能な暗号では，A から B への再暗号化鍵 $rk_{A\to B}$ を用いることで，プロキシは A 宛ての暗号文を平文に復号することができないにもかかわらず，B 宛ての暗号文に変換することができる．

2.2 双線形写像

写像 $e : \mathbb{G}_0 \times \mathbb{G}_1 \to \mathbb{G}_2$ が以下の三つを満たすとき，e を双線形写像という．

（1） $\mathbb{G}_0$ と $\mathbb{G}_1$ は位数が同じ素数 p の群

（2） $\forall a, b \in \mathbb{Z}_p, g \in \mathbb{G}_0, h \in \mathbb{G}_1$ に対して，$e(g^a, h^b) = e(g, h)^{ab}$ であり，効率的に計算可能

（3） 非縮退性を持つ．つまり，g と h がそれぞれ $\mathbb{G}_0, \mathbb{G}_1$ の原始元ならば，$e(g, h)$ は $\mathbb{G}_2$ の原始元である

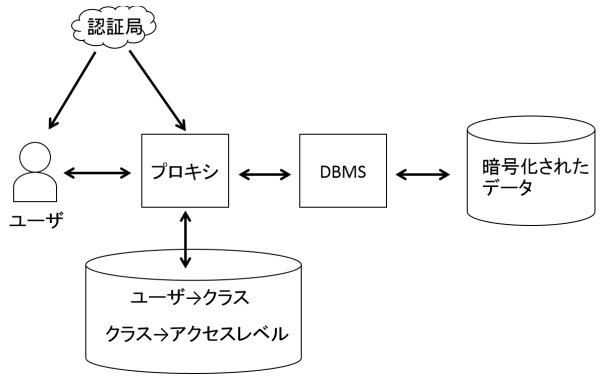

図 1　Acolcoci のモデル

3. 関 連 研 究

本節では，本研究の基となる研究を紹介する．

3.1 概　　要

児玉らは，プロキシ再暗号化可能な暗号を用いることで，柔軟なアクセス制御を行いながらも従来よりも少ない計算量でユーザのアクセス権限削除を可能にする Acolcoci と呼ばれる手法を提案した [4]．Acolcoci のモデルを図 1 に示す．

Giereth の手法 [5] など，従来の暗号化データの共有方法では共有グループからユーザをリボケーションする際，共有データを暗号化しなおし，復号するための新たな鍵を再配布する必要がある．

Acolcoci では，プロキシサーバによるプロキシ再暗号化を通じてデータにアクセスするため，再暗号化鍵を破棄するだけでデータへのアクセスができなくなり，ユーザの速やかなリボケーションが可能である．

さらに従来のアクセスレベルによるアクセス制御に加え，個人間の関係を利用したアクセス制御を導入した．前者をオープンアクセス，後者をクローズドアクセスと呼ぶ．

3.2 Acolcoci で利用する暗号

Acolcoci では Blaze らのプロキシ再暗号化可能な暗号である BBS 暗号 [3] を利用している．BBS 暗号について簡単に述べる．

セットアップ

安全素数 $p = 2q + 1$，$\mathbb{Z}_p^*$ の原始元 g を決め，公開パラメータとする．

鍵生成

$a \leftarrow \mathbb{Z}_{2q}^*$ をランダムに選び，$sk_A = a, pk_A = g^a \pmod p$ をそれぞれ A の秘密鍵，公開鍵とする．

再暗号化鍵生成

sk_A, sk_B を入力として，$rk_{A\to B} = \frac{sk_B}{sk_A} = \frac{b}{a}$ を A 宛ての暗号文から B 宛ての暗号文に変換する再暗号化鍵とする．

暗号化

平文 $m \in \mathbb{Z}_{2q}^*$ を pk_A で次のように暗号化した暗号文 c_A を A 宛ての暗号文とする．

$$c_A = (mg^s, (pk_A)^s) = (mg^s, g^{as}) \pmod p$$

再暗号化

c_A と $rk_{A\to B}$ を入力として，以下のように c_B へと再暗号化を行う．$c_B = (mg^s, (pk_A^s)^{rk_{A\to B}}) = (mg^s, (g^{as})^{\frac{b}{a}}) = (mg^s, g^{bs}) (= (mg^s, (pk_B)^s) \pmod p)$

復号

c_A を sk_A により以下のように復号する．$\frac{mg^s}{(g^{as})^{sk_A^{-1}}} = \frac{mg^s}{(g^{as})^{a^{-1}}} = m \pmod p$

4. 先行研究の課題

本節では，3. 節で述べた研究 [4] に残された課題と本研究の目的を述べる．

4.1 結託によってデータの機密性が失われる

ユーザとプロキシの結託により，ユーザに割り当てられたアクセスレベルを超える任意のデータにアクセスできる．これには三つの原因があり，それぞれ別々の攻撃方法となる．

- 原因 1：BBS 暗号が双方向性を持つ

双方方向性とは，暗号文 c を c' に再暗号化できる権利のみから c' を c に変換可能である性質をいう．

Acolcoci では，データを蓄積する際ユーザは任意のポリシー，つまりアクセスレベルを指定できるので，認証局は任意のアクセスレベル l に対して $rk_{A\to l}$ を作成する．したがって，BBS 暗号の双方向性により，アクセス権限を持たない l に対しても $rk_{l\to A}$ を得ることができ，任意のデータを A 宛ての暗号文に再暗号化可能である．

- 原因 2：BBS 暗号が双方向性を持ち，かつ耐結託性を持たない

プロキシ再暗号化において耐結託性とは，sk_A または sk_B と $rk_{A\to B}$ から sk_B または sk_A が得られないことをいう．

BBS 暗号では，ユーザ A の秘密鍵 sk_A と，プロキシが持つ A の再暗号化鍵 $rk_{A\to l} = \frac{sk_l}{sk_A}$ から，$sk_A \times rk_{A\to l} = sk_A \times \frac{sk_l}{sk_A} = sk_l$ とすることでアクセスレベル l に紐付けられた秘密鍵を得ることができる．データを蓄積する際，ユーザは任意のアクセスレベルを指定できるので，すべてのアクセスレベルへの再暗号化鍵が存在する．したがって双方向性とから，任意のアクセスレベルに対応する秘密鍵をすべて得ることができ，任意の暗号文を復号可能である．

- 原因 3：暗号化の際，本来乱数であるべき箇所を定数にしてすべてのユーザで共有している

これは検索方法に起因する．Acolcoci における検索の仕組みを図 2 に示す．

本来 s は乱数なので，データベースに保存した後から暗号文同士の一致検索が行えない．そのため乱数を固定してすべてのユーザで共有することで一致検索を可能にしている．その結果，データベース中の暗号文を直接復号する秘密鍵や再暗号化鍵を知らずとも，g^s を計算し，データ m をマスクしている要素を除算することであらゆるデータを復号できる．つまり，結託しているユーザから s を知ることで，プロキシに送られてくるすべての暗号文を復号可能である．

また，ユーザと DBA の結託によっても任意のデータにアクセスできる（図 3）．

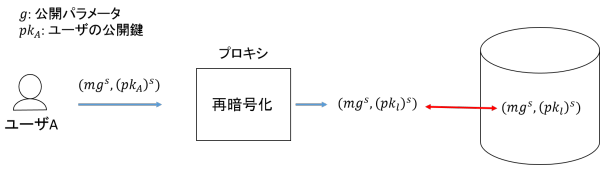

図 2　検索の仕組み

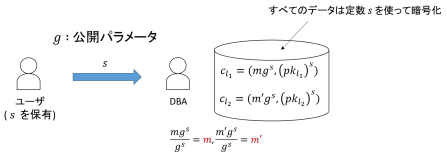

図 3　ユーザと DBA の結託による任意のデータへのアクセス

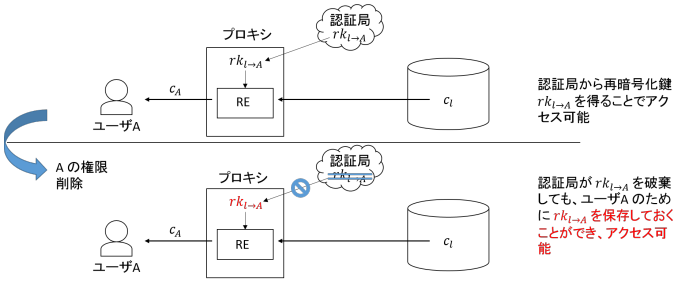

図 4　結託ユーザの権限削除

これは前述したように，暗号化の際に乱数ではなく全ユーザ共通の定数を使っているためである．

4.2　結託ユーザのリボケーションができない

単独のユーザのリボケーションは可能であるが，プロキシまたは DBA と結託しているユーザのリボケーションができない．

これには二つの原因がある．

一つ目の原因は，再暗号化鍵を使わずにデータにアクセスできるためである．これは，前述した課題の原因 2, 3 に起因する．

二つ目の原因は，ユーザとプロキシが結託していると，当該ユーザの再暗号化鍵 $rk_{l\to A}$ を保存しておけばいつでもデータにアクセスできるためである（図 4）．

4.3　範囲クエリに対応できない

一般的な暗号化では暗号文からは元の平文の大小比較ができないため，検索はクエリとして出した暗号文とデータベースに保存されている暗号文が同じか否かを判定できれば処理可能な一致検索のみサポートできる．

4.4　個人間の関係を利用したアクセス制御が結託に対してセキュアでない

これはオープンアクセスで用いる暗号化とクローズドアクセスで用いる暗号化が同じであることに起因する．

クローズドアクセスの例を図 5 に示す．

アクセスレベルの階層は $l_1 > l_2$ とする．アクセスレベルの観点からは Bob は Alice のデータ m にアクセスできないが，Alice が自分の家族にはアクセスを許可しているので Bob はアクセス可能である．一方 Carol はアクセスできない．

Acolcoci ではいかなる場合も，アクセスレベルに紐付けられた公開鍵による暗号文を再暗号化する再暗号化鍵を作ることで

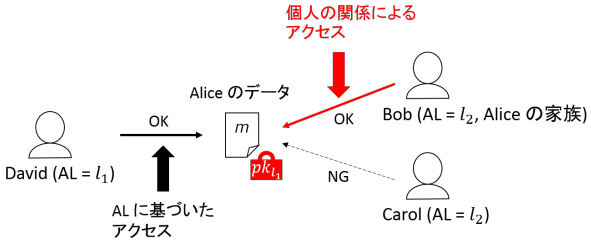

図 5　クローズドアクセス

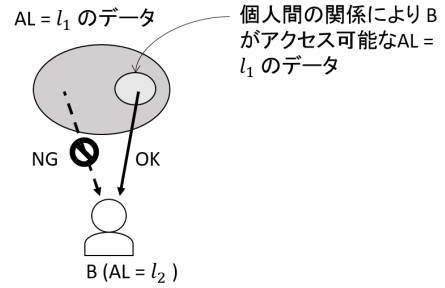

図 6　クローズドアクセスによりアクセス可能となるべきデータ

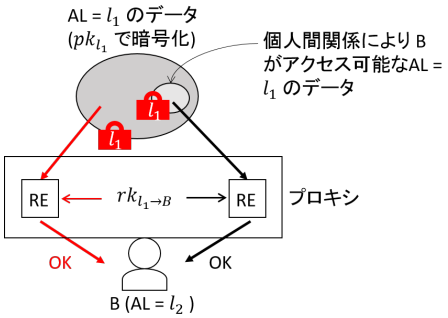

図 7　Acolcoci にてクローズドアクセスによりアクセス可能となるデータ

暗号文の復号を行う．そのため，プロキシと結託することでアクセス権限のないデータにもアクセス可能である．

図 5 の例でいうと，l_1 に設定されたデータはすべて l_1 に紐づいた鍵で暗号化されているため，m へアクセスするためには $rk_{l_1\to Bob}$ を認証局が発行し，プロキシが Bob が復号可能な暗号文に再暗号化することでアクセスを可能にする．本来，図 6 に示すように，アクセスを許されるのは l_1 に設定されているデータの中の一部に過ぎないが，$rk_{l_1\to Bob}$ によってプロキシが再暗号化することで l_1 に設定されたすべてのデータの暗号文を Bob が復号できる（図 7）．

5.　提 案 手 法

5.1　研 究 目 的

本研究では，4. 節で述べた課題を解決することで，以下の性質を持つシステムを実現することを目的とする．

- 二者間の結託に対してデータの機密性を保つ
- 結託ユーザであっても効率的なリボケーションが可能
- 範囲クエリに対応可能
- 個人間の関係を利用したアクセスにおける耐結託

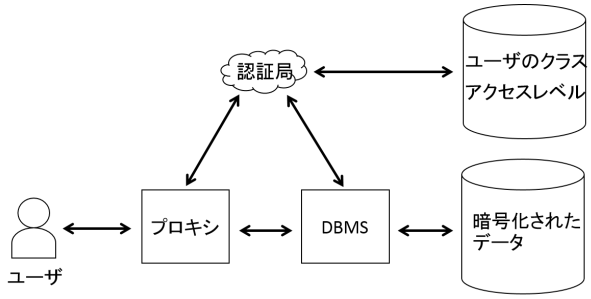

図 8　提案手法のモデル

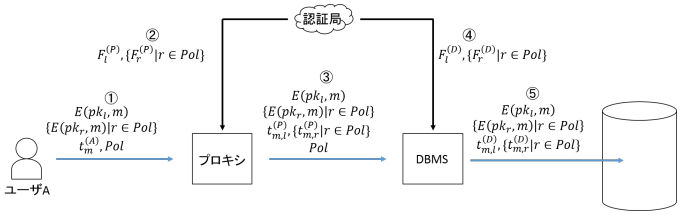

図 9　データの蓄積

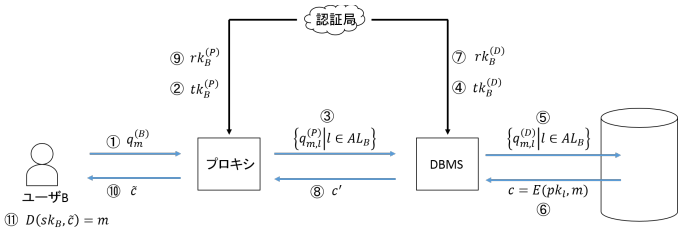

図 10　オープンアクセスによるデータの検索

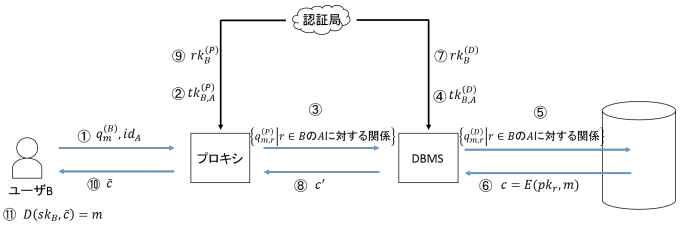

図 11　クローズドアクセスによるデータの検索

5.2　提案手法の概要

本研究でも先行研究 [4] 同様，アクセスレベルによるアクセス制御と個人間の関係を利用したアクセス制御の二種類のアクセス制御方法を行い，前者をオープンアクセス，後者をクローズドアクセスと呼ぶ．

本研究で想定するモデルを図 8 に示す．

ユーザは高々一つのクラスに所属し，クラスは高々1 つのアクセスのレベル（アクセスレベル）を割り当てられ，これらの情報は認証局が保持する．

アクセスレベルは階層関係を持つ．つまり，あるアクセスレベル l を割り当てられたクラスに所属するユーザは，l 以下のアクセスレベルを対応付けられたデータへのアクセスが可能である．

ユーザはプロキシサーバに対して，保有するデータの蓄積を要求することと，蓄積されているデータの検索を要求することができる．

図 9 に情報の蓄積の流れを，図 10 と図 11 にオープンアクセスとクローズドアクセスによるデータの検索の流れを示す．

5.3　データの暗号化と復号

4.1 項で述べた，一つ目の課題の原因 1 と 2 を解決するため，単方向性を持つプロキシ再暗号化可能な暗号 [6] を利用する．

本研究では，プロキシまたは DBA と結託しているユーザであっても速やかなリボケーションを可能にするために，プロキシだけでなく DBMS が復号に関与するように再暗号化鍵に改変を加える．

また，クローズドアクセスは個人間の関係一つひとつを公開鍵に紐付けることで実現する．

以下に具体的な方法を示す．

セットアップ

H をハッシュ関数，$\mathbb{G}, \mathbb{G}_T$ を位数が素数 p の巡回群，g を $\mathbb{G}$ の原始元，$e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ を双線形写像，$(g, Z = e(g, g))$ をシステムパラメータとする．

すべてのアクセスレベル l について $t_l \leftarrow \mathbb{Z}_p$ をランダムに選び，$pk_l = g^{t_l}$ をアクセスレベル l に紐付いた公開鍵とする．

$b, b' \leftarrow \mathbb{Z}_p$ をランダムに選び，ユーザ B の秘密鍵を $sk_B = b$, DBMS 用再暗号化鍵を $rk_B^{(D)} = b'^{-1}$ とする．また，B のオープンアクセスにおけるプロキシ用再暗号化鍵を $rk_B^{(P)} = \{(H(l), g^{\frac{b'b}{t_l}}) | l \in AL_B\}$ とする．ここで，AL_B は B がアクセス可能なレベルの集合を表す．

暗号化

どのアクセスレベル（l），またどの関係（$r_1, \ldots, r_n$）のユーザにアクセスを許可するかを定めるポリシー $\mathrm{Pol} = \{l, r_1, \ldots, r_n\} (n \geqq 0$：アクセスを許可する関係の個数) を決める．関係とは，例えば「A の家族」といった関係である．

次に，アクセスレベル l に対応する暗号文を，乱数 $s \leftarrow \mathbb{Z}_p$ を選び，$(H(l), mZ^s, pk_l^s)$ とする．暗号化は [6] と同じだが，本研究では，どのアクセスレベルで暗号化しているか判別できるよう，l のハッシュ値を含める．

続いて，個人間の関係 $r \in \mathrm{Pol}$ に対応する暗号文を同様に作成する．つまり，$\forall r \in \mathrm{Pol}$ について乱数 $s \leftarrow \mathbb{Z}_p$ を選び，$(H(r), mZ^s, pk_r^s)$ を関係 r に対応する暗号文とする．但し，暗号化を行う時点で pk_r が存在しない場合，このとき認証局が pk_r を作成する．すなわち，$t_r \leftarrow \mathbb{Z}_p$ をランダムに選び，$pk_r = g^{t_r}$ とする．

次に暗号文の復号について述べる．

復号は DBMS による暗号文の再暗号化とプロキシによるさらなる再暗号化を経てユーザが復号可能な暗号文を生成し，それをユーザが復号することによって行う．

DBMS **による再暗号化**

まず，認証局からユーザ B の DBMS 用再暗号化鍵 $rk_B^{(D)} = b'^{-1}$ を取得する．

オープンアクセスの場合，$c' = (mZ^s, (pk_l^s)^{rk_B^{(D)}}) = (mZ^s, g^{\frac{t_l s}{b'}})$ を変換後の暗号文としてプロキシに送る．

クローズドアクセスの場合，$c' = (mZ^s, (pk_r^s)^{rk_B^{(D)}}) = (mZ^s, g^{\frac{t_r s}{b'}})$ を変換後の暗号文としてプロキシに送る．

プロキシによる再暗号化

オープンアクセスの場合，認証局からオープンアクセスにおけるユーザ B のプロキシ用再暗号化鍵

$$rk_B^{(P)} = \{(H(l), g^{\frac{b'b}{t_l}}) | l \in AL_B\}$$

を取得し，以下のように再暗号化する．まず

$$e(g^{\frac{t_l s}{b'}}, rk_B^{(P)}) = e(g^{\frac{t_l s}{b'}}, g^{\frac{b'b}{t_l}}) = e(g,g)^{bs} = Z^{bs}$$

を計算し，$\tilde{c} = (mZ^s, Z^{bs})$ を変換後の暗号文として B に送る．

クローズドアクセスの場合，認証局からユーザ B のユーザ A に対する再暗号化鍵

$$rk_{B,A}^{(P)} = \{(H(r), g^{\frac{b'b}{t_r}}) | r : \text{B と A の間の関係 }\}$$

を取得し，オープンアクセスの場合と同様にして再暗号化を行う．つまり，

$$e(g^{\frac{t_r s}{b'}}, rk_B^{(P)}) = e(g^{\frac{t_r s}{b'}}, g^{\frac{b'b}{t_r}}) = e(g,g)^{bs} = Z^{bs}$$

を計算し，$\tilde{c} = (mZ^s, Z^{bs})$ を変換後の暗号文として B に送る．但し，再暗号化を行う時点で $rk_{B,A}^{(P)}$ が存在しない場合は次のようにして認証局が作成する．B の A に対するすべての関係 r に対して，$t_r \leftarrow \mathbb{Z}_p$ をランダムに選び，$rk_{B,A}^{(P)} = \{(H(r), g^{\frac{b'b}{t_r}}) | \forall r : \text{B の A に対する関係 }\}$ とする．

ユーザによる復号

ユーザによる復号はオープンアクセスもクローズドアクセスも同じで，

$$\frac{mZ^s}{(Z^{bs})^{sk_B^{-1}}} = \frac{mZ^s}{(Z^{bs})^{b^{-1}}} = m$$

として復号する．

5.4 検索タグの作成

この節では検索タグの作成方法を述べる．

検索タグと検索に使用する落とし戸は Lewi らの順序比較可能暗号 [7] に基づく．この暗号では，平文 $x = x_1 x_2 \ldots x_n$ の左暗号文 $ct_L^{(x)}$ と平文 $y = y_1 y_2 \ldots y_n$ の右暗号文 $ct_R^{(y)}$ から x と y の一致比較・大小比較が可能である．セキュリティに関しては，$ct_L^{(x)}, ct_R^{(y)}$ から x, y の大小と，$x_i \neq y_i$ となる最小のブロックのインデックス i 以外の情報は漏れない．

セキュリティパラメータ $\lambda \in \mathbb{N}$，メッセージ空間の大きさ $N > 0$，整数 $d, n > 0 (d^n \geqq N)$ を固定する．$F : \{0,1\}^\lambda \times \mathbb{Z}_N \to \{0,1\}^\lambda, F^{(P)} : \{0,1\}^\lambda \times \mathbb{Z}_N \to \{0,1\}^\lambda, F^{(D)} : \{0,1\}^\lambda \times \mathbb{Z}_N \to \{0,1\}^\lambda$ をそれぞれ，ユーザ，プロキシ，DBMS 用の擬似ランダム関数（PRF）[8]，$H : \{0,1\}^\lambda \times \{0,1\}^\lambda \to \mathbb{Z}_3$ をハッシュ関数，$\pi : \{0,1\}^\lambda \times \mathbb{Z}_d \to \mathbb{Z}_d$ を擬似ランダム置換（PRP）とする．$x_1 x_2 \ldots x_n$ を，x のビット表現を n 等分し，十進数で表した値 $x_i (i = 1, \ldots, n)$ を用いた表現とし，$x = x_1 x_2 \ldots x_n$ に対し，$x_{|i} = x_1 x_2 \ldots x_i$ すなわち先頭から i 番目までのプレフィックスとする．但し，$x_{|0}$ は空のプレフィックスとする．また，$a||b$ は a, b のビット連結を表す．さらに，a, b の大小を判定する関数 $\mathrm{CMP}(a, b)$ を次のように定義する．

$$\mathrm{CMP}(a,b) = \begin{cases} 2 & (a < b) \\ 0 & (a = b) \\ 1 & (a > b) \end{cases}$$

ユーザが共有する PRF や各アクセスレベル，個人間の関係に紐付けられる PRF は次のように認証局が作成する．$k_1, k_2 \leftarrow \{0,1\}^\lambda$ をランダムに選び，$F(k_1, \cdot), F(k_2, \cdot)$ をユーザ用の PRF とする．$k_{1,l}^{(P)}, k_{2,l}^{(P)}, k_{1,r}^{(P)}, k_{2,r}^{(P)} \leftarrow \{0,1\}^\lambda$ をランダムに選び，$F(k_{1,l}^{(P)}, \cdot), F(k_{2,l}^{(P)}, \cdot)$ をアクセスレベル l に紐付いたプロキシ用の PRF とし，それぞれ $F_{1,l}^{(P)}(\cdot), F_{1,2}^{(P)}(\cdot)$ と書く． $F(k_{1,r}^{(P)}, \cdot), F(k_{2,r}^{(P)}, \cdot)$ を個人間関係 r に紐付けられたプロキシ用の PRF とし，それぞれ $F_{1,r}^{(P)}(\cdot), F_{2,r}^{(P)}(\cdot)$ と書く．同様に，$k_{1,l}^{(D)}, k_{2,l}^{(D)}, k_{1,r}^{(D)}, k_{2,r}^{(D)} \leftarrow \{0,1\}^\lambda$ をランダムに選び，$F(k_{1,l}^{(D)}, \cdot), F(k_{2,l}^{(D)}, \cdot)$ をアクセスレベル l に紐付いた DBMS 用 PRF とし，それぞれ $F_{1,l}^{(D)}(\cdot), F_{2,l}^{(D)}(\cdot)$ と書く．また，$F(k_r^{(D)}, \cdot), F(k_r^{(D)}, \cdot)$ を， 個人間関係 r に紐付いた DBMS 用 PRF とし，それぞれ $F_{1,r}^{(D)}(\cdot), F_{2,r}^{(D)}(\cdot)$ と書く．

タグの作成方法を図 9 の流れに沿って示す．

（1） ユーザによるタグの作成

乱数 $r \leftarrow \{0,1\}^\lambda$ をランダムに選ぶ．すべての $i \in \{1, 2, \ldots, n\}, j \in \mathbb{Z}_d$ に対し，$j^* = \pi^{-1}(F_2(y_{|i-1}), j)$ を計算し，

$$z_{ij} = CMP(j^*, y_i) + H(F_1(y_{i-1}||j), r) \,(\mathrm{mod}\, 3)$$

とする．$t_y^{(A)} = (r, v_1, v_2, \ldots, v_n)$ （$v_i = (z_{i1}, z_{i2}, \ldots, z_{id})$）と $\tilde{y} = \tilde{y}_1 \tilde{y}_2, \ldots, \tilde{y}_n \, (\tilde{y}_i = \pi(F_2(y_{|i-1}), y_i))$ ，Pol をプロキシに送る．

（2） ユーザ作成タグを変換するためのプロキシ用擬似ランダム関数の取得

Pol を見て，アクセスレベル l に紐付けられたプロキシ用擬似ランダム関数 $F_{1,l}^{(P)}, F_{2,l}^{(P)}$，個人間関係 r に紐付けられたプロキシ用擬似ランダム関数 $F_{1,r}^{(P)}, F_{2,r}^{(P)}$ を認証局から取得する．

（3） ユーザ作成タグの変換

$$w_{ik} = z_{i\pi^{-1}(F_{2,l}^{(P)}(\tilde{y}_{|i-1}),k)} + H(F_{1,l}^{(P)}(\tilde{y}_{|i-1}||k), r) \,(\mathrm{mod}\, 3)$$

を計算し，アクセスレベル l に対応するプロキシ作成タグを

$$t_{y,l}^{(P)} = (r, w_{11}, \ldots, w_{1d}, w_{21}, \ldots, w_{2d}, \ldots, w_{n1}, \ldots, w_{nd})$$

とする．次に，$\forall r \in$ Pol に対して

$$w_{ik} = z_{i\pi^{-1}(F_{2,r}^{(P)}(\tilde{y}_{|i-1}),k)} + H(F_{1,r}^{(P)}(\tilde{y}_{|i-1}||k), r) \,(\mathrm{mod}\, 3)$$

を計算し，関係 r に対応するプロキシ作成タグを

$$t_{y,r}^{(P)} = (r, w_{11}, \ldots, w_{1d}, w_{21}, \ldots, w_{2d}, \ldots, w_{n1}, \ldots, w_{nd})$$

とする．プロキシ作成タグ $t_{y,l}^{(P)}, \{t_{y,r}^{(P)} | r \in \text{Pol}\}$ ， $\hat{y}_l, \{\hat{y}_r | r \in \text{Pol}\}$ を DBMS に送る．但し，

$$\hat{y}_l = \hat{y}_1 \hat{y}_2 \cdots \hat{y}_n \, (\hat{y}_i = \pi(F_{2,l}^{(P)}(\tilde{y}_{|i-1}), \tilde{y}_i))$$

$$\hat{y}_r = \hat{y}_1 \hat{y}_2 \cdots \hat{y}_n \, (\hat{y}_i = \pi(F_{2,r}^{(P)}(\tilde{y}_{|i-1}), \tilde{y}_i))$$

である．

（4） プロキシ作成タグを変換するための DBMS 用擬似ランダム関数の取得

Pol を見て，アクセスレベル l に紐付けられた DBMS 用擬似ランダム関数 $F_{1,l}^{(D)}, F_{2,l}^{(D)}$，個人間関係 r に紐付けられた DBMS 用擬似ランダム関数 $F_{1,r}^{(D)}, F_{2,r}^{(D)}$ を認証局から取得する．

（5） プロキシ作成タグの変換

$$s_{im} = w_{i\pi^{-1}(F_{2,l}^{(D)}(\hat{y}_{|i-1}),m)} + H(F_{1,l}^{(D)}(\hat{y}_{|i-1}||m), r) \,(\mathrm{mod}\, 3)$$

を計算し，アクセスレベル l に対応する DBMS 作成タグを

$$t_{y,l}^{(D)} = (r, s_{11}, \ldots, s_{1d}, s_{21}, \ldots, s_{2d}, \ldots, s_{n1}, \ldots, s_{nd})$$

とする．次に，$\forall r \in$ Pol に対して

$$s_{im} = w_{i\pi^{-1}(F_{2,r}^{(D)}(\hat{y}_{|i-1}),m)} + H(F_{1,r}^{(D)}(\hat{y}_{|i-1}||m), r) \,(\mathrm{mod}\, 3)$$

を計算し，関係 r に対応する DBMS 作成タグを

$$t_{y,r}^{(D)} = (r, s_{11}, \ldots, s_{1d}, s_{21}, \ldots, s_{2d}, \ldots, s_{n1}, \ldots, s_{nd})$$

とする．$t_{y,l}^{(D)}, \{t_{y,r}^{(D)} | r \in \mathrm{Pol}\}$ を最終的な検索可能タグとしてデータベースに保存する．

5.5 タグの検索

タグの検索は，Lewi らの順序比較可能暗号 [7] における左暗号に基づく．クローズドアクセスの場合はオープンアクセスの場合とほとんど同じなので，ここではオープンアクセスの場合だけを図 10 に沿って述べる．

（1） ユーザによる落とし戸の作成

$\forall i \in \{1, \ldots, n\}$ に対して，$\tilde{x}_i = \pi(F_2(x_{|i-1}), x_i), F_1(x_{|i-1}||\tilde{x}_i)$ を計算し，

$$q_x^{(B)} = \begin{bmatrix} (F_1(x_{|0}||\tilde{x}_1), \tilde{x}_1) \\ (F_1(x_{|1}||\tilde{x}_2), \tilde{x}_2) \\ \vdots \\ (F_1(x_{|n-1}||\tilde{x}_n), \tilde{x}_n) \end{bmatrix}$$

をユーザ作成落とし戸として，プロキシに送る．

（2） 落とし戸を変換するプロキシ用変換鍵の取得

オープンアクセスにおける B に紐付けられたプロキシ用落とし戸変換鍵 $tk_B^{(P)} = \{(F_{1,l}^{(P)}, F_{2,l}^{(P)}) | l \in AL_B\}$ を認証局から取得する．

（3） プロキシによる落とし戸の変換

$q_x^{(B)}$ から B がアクセス可能な各アクセスレベルに対応する落とし戸を作る．$\forall i \in \{1, \ldots, n\}, \forall l \in AL_B$ に対して，$\hat{x}_i = \pi(F_{2,l}^{(P)}(\tilde{x}_{|i-1}), \tilde{x}_i), F_{1,l}^{(P)}(\tilde{x}_{|i-1}||\hat{x}_i)$ を計算し，

$$q_{x,l}^{(P)} = \begin{bmatrix} (F_1(x_{|0}||\tilde{x}_1), F_{1,l}^{(P)}(\tilde{x}_{|0}||\hat{x}_1), \hat{x}_1) \\ (F_1(x_{|1}||\tilde{x}_2), F_{1,l}^{(P)}(\tilde{x}_{|1}||\hat{x}_2), \hat{x}_2) \\ \vdots \\ (F_1(x_{|n-1}||\tilde{x}_n), F_{1,l}^{(P)}(\tilde{x}_{|n-1}||\hat{x}_n), \hat{x}_n) \end{bmatrix}$$

とする．変換した落とし戸の集合 $Q_x^{(P)} = \{q_{x,l}^{(P)} | l \in AL_B\}$ を DBMS へ送る．

（4） 落とし戸を変換する DBMS 用変換鍵の取得

オープンアクセスにおける Bob に紐付けられた DBMS 用落とし戸変換鍵 $tk_B^{(D)} = \{(F_{1,l}^{(D)}, F_{2,l}^{(D)}) | l \in AL_B\}$ を認証局から取得する．

（5） DBMS による落とし戸の変換とクエリの実行

$tk_B^{(D)}, Q_x^{(P)}$ から B がアクセス可能な各アクセスレベルに対応する最終的な落とし戸を作る．$\forall i \in \{1, \ldots, n\}, \forall l \in AL_B$ に対して　$\check{x}_i = \pi(F_{2,l}^{(D)}(\hat{x}_{|i-1}), \hat{x}_i), F_{1,l}^{(D)}(\hat{x}_{|i-1}||\check{x}_i)$ を計算し，

$$q_{x,l}^{(D)} = \begin{bmatrix} (F_1(x_{|0}||\tilde{x}_1), F_{1,l}^{(P)}(\tilde{x}_{|0}||\hat{x}_1), F_{1,l}^{(D)}(\hat{x}_{|0}||\check{x}_1), \check{x}_1) \\ (F_1(x_{|1}||\tilde{x}_2), F_{1,l}^{(P)}(\tilde{x}_{|1}||\hat{x}_2), F_{1,l}^{(D)}(\hat{x}_{|1}||\check{x}_2), \check{x}_2) \\ \vdots \\ (F_1(x_{|n-1}||\tilde{x}_n), F_{1,l}^{(P)}(\tilde{x}_{|n-1}||\hat{x}_n), F_{1,l}^{(D)}(\hat{x}_{|n-1}||\check{x}_n), \check{x}_n) \end{bmatrix}$$

とする．変換した落とし戸の集合 $Q_x^{(D)} = \{q_{x,l}^{(D)} | l \in AL_B\}$ を用いてデータベースに保存されているタグと一致比較，大小比較を次のように行う．

$\forall q_{x,l}^{(D)} \in Q_x^{(D)}$ について，$s_{i\tilde{x}_i} - H(F_1(x_{|i-1}||\tilde{x}_i), r) - H(F_{1,l}^{(P)}(\tilde{x}_{|i-1}||\hat{x}_i), r) - H(F_{1,l}^{(D)}(\hat{x}_{|i-1}||\check{x}_i), r) \neq 0 \,(\mathrm{mod}\, 3)$ となる最小の $i = u$ を見つけ，$s_{u\tilde{x}_u} - H(F_1(x_{|u-1}||\tilde{x}_u), r) - H(F_{1,l}^{(P)}(\tilde{x}_{|u-1}||\hat{x}_u), r) - H(F_{1,l}^{(D)}(\hat{x}_{|u-1}||\check{x}_u), r) = \mathrm{CMP}(x, y) \,(\mathrm{mod}\, 3)$ を出力することによって行う．そのような u が存在しなければ，0 を出力する．

6. セキュリティに関する考察

本研究では，信頼できないプロキシと DBA からデータの機密性 (confidentiality) を守ることをセキュリティ上の目的とし，データの完全性 (integrity) や可用性（availability）は考慮しない．つまり，プロキシや DBA はセッション中のデータやデータベース中のデータ，クエリ，クエリの結果を改竄または削除するといった攻撃が可能であるが，これらは popa らの研究 [9] などと同様，本研究の対象外とする．

また，二者間の結託のみ考慮する．すなわちユーザ，プロキシ，DBA 三者の結託による攻撃は想定しない．

以降では，データベースに保存されている，アクセスレベル l に対応する暗号文 $c_l = (mZ^s, g^{t_l s})$ を直接復号する鍵を sk_l とする．

暗号文 c_l は，$g^{t_l^{-1}}$ によって次のように直接復号できる．

（1） $e(g^{t_l s}, g^{t_l^{-1}}) = e(g, g)^s = Z^s$

（2） $mZ^s / Z^s = m$

したがって，$sk_l = g^{t_l^{-1}}$ である．

データの機密性を守るだけでなく，ユーザが出すクエリの内容がプロキシと DBA に対して隠される秘匿検索が可能であることが望ましいので，秘匿検索についても考察する．

図 12 に想定する脅威モデルを示す．

以降では，データベース中の暗号文を直接復号するための鍵を sk とする．

6.1 脅威 1: プロキシ単独

信頼できないプロキシによるデータへの不正なアクセスと，外部からのプロキシへの攻撃を含む．プロキシは暗号文を復号

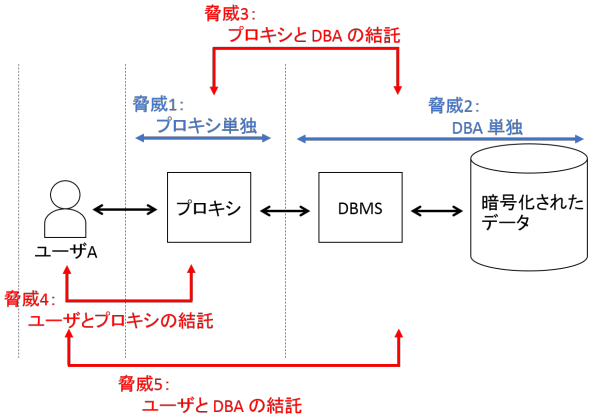

図 12　脅威モデル

するための鍵を持つことはできないので，いかなるデータにもアクセスできない．タグや落とし戸もまた，順序比較可能暗号 [7] の安全性から元のデータの秘匿性が保たれ，秘匿検索も可能である．

また，先行研究同様，再暗号化鍵を破棄するだけでリボケーションが可能である．

6.2　脅威 2: DBA 単独

DBA によるデータへの不正なアクセスと，外部からの DBMS への攻撃を含む．脅威 2 も脅威 1 と同様の理由からデータの秘匿性が保たれ，さらに秘匿検索が可能である．

また，先行研究同様，再暗号化鍵を破棄するだけでリボケーションが可能である．

6.3　脅威 3: プロキシと DBA の結託

プロキシと DBA の結託によるデータへの不正アクセスである．暗号文とタグそれぞれについて考察する．

6.3.1　暗　号　文

データベースに保存されている暗号文を復号する鍵を得られないことが必要なので，この観点から考察する．

認証局からプロキシはユーザ U の再暗号化鍵 $rk_U = g^{\frac{u'u}{t_l}}$，DBA は $sk_U^{(D)} = u'^{-1}$ をそれぞれ得られるので，$g^{\frac{u}{t_l}}$ を計算可能である．しかし，u が不明なため $g^{t_l^{-1}}$ を知ることはできず，データベース中の暗号文を復号することはできない．

6.3.2　検 索 タ グ

プロキシと DBMS では，ユーザから受け取ったタグを置換し，誤差を加える操作をすだけなので，順序比較可能暗号 [7] の安全性からデータの秘匿性は守られる．

6.3.3　秘匿検索について

検索タグ同様，プロキシと DBMS ではユーザから受け取った落とし度を置換し，新たな乱数を付加するだけなので，順序比較可能暗号 [7] の安全性からこの攻撃に対して安全であり，秘匿検索が可能である．

6.3.4　リボケーションについて

このケースではユーザとは結託しないので，効率的なリボケーションが可能である．

6.4　脅威 4: ユーザとプロキシの結託

ユーザとプロキシの結託によるデータへの不正アクセスである．

6.4.1　暗　　号

4.1 項で，ユーザとプロキシの結託によって任意のデータにアクセスできる原因を三つ挙げた．それらが解消されているかどうかを考える．

本研究で利用する改良型プロキシ再暗号化 [6] が双方向性を持たないため，双方向性を持たない．したがって原因 1 と 2 は解消される．

原因 3 も，本研究では検索タグを用いた検索を行うことで暗号化には乱数を使うことができ，定数を共有することはないので解消される．

したがって，結託しても，ユーザは割り当てられたアクセスレベルを超えるデータにはアクセスできない．

6.4.2　検 索 タ グ

提案手法では，ユーザはデータ y のタグの作成時に $\tilde{y} = \tilde{y}_1\tilde{y}_2 \ldots \tilde{y}_n\,(\tilde{y}_i = \pi(F_{2,l}^{(P)}(y_{|i-1}), y_i))$ をプロキシに送る必要がある．

$y \neq y'$ である任意の平文 y, y' に対して，y と y' は i 番目で初めて異なるとする．$y_{|i-1} = y'_{|i-1} = s$ とすると，

$$\tilde{y}_i = \pi(F_{2,l}^{(P)}(y_{|i-1}), y_i) = \pi(F_{2,l}^{(P)}(s), y_i)$$

$$\tilde{y'}_i = \pi(F_{2,l}^{(P)}(y'_{|i-1}), y'_i) = \pi(F_{2,l}^{(P)}(s), y'_i)$$

であり，$y_i \neq y'_i$ とから，$\tilde{y} \neq \tilde{y'}$ が成り立つ．

y から $\tilde{y}$ への写像が単射であることと，ユーザと結託したプロキシは任意のデータ y と $\tilde{y}$ のペアを得ることができることから，任意の y と $\tilde{y}$ の対応表を作ることができる．テーブルを参照することにより，任意の結託していないユーザから受け取る $\tilde{y^*}$ から元の平文 y^* を得ることができる．

しかし，メッセージ空間は十分大きく，このような総当たり攻撃はコストが大きく現実的ではないと考えられるので，本研究では考慮しない．したがって，順序比較可能暗号 [7] の安全性から，データの機密性は保たれる．

6.4.3　秘匿検索について

落とし戸に関してもタグ同様，x とユーザが作成する落とし戸に含まれる $\tilde{x}$ の対応表が作れるが，検索タグと同じ理由から秘匿検索が可能である．

6.4.4　リボケーションについて

で述べたように，プロキシと結託しているユーザのリボケーションができない原因は，プロキシが再暗号化鍵を保存できることと，再暗号化鍵がなくともデータにアクセスできるためであった．

提案手法では，DBMS が復号に参加しており，DBMS が鍵を破棄することで一つ目の原因は解消される．二つ目の原因について考える．

プロキシと結託しているユーザ U の秘密鍵 $sk_U = u$ と，U の再暗号化鍵 $rk_{U\to l}^{(P)} = g^{\frac{uu'}{t_l}}$ から $g^{\frac{u'}{t_l}}$ が得られるが，u' が不明なため $sk_l = g^{t_l^{-1}}$ を得ることはできない．また，検索タグによる検索を行うため定数を共有することはない．

したがって，プロキシと結託しているユーザでも効率的なリボケーションが可能である．

ユーザとの結託に耐性を持つ

	脅威1 プロキシ単独	脅威2 DBA 単独	脅威3 プロキシ＋DBA	脅威4 ユーザ＋プロキシ	脅威5 ユーザ＋DBA
提案手法	○	○	○	○	○
先行研究[4]	○	○	○	×	×

図 13　提案手法と先行研究の機密性に関する比較

6.5　脅威 5: ユーザと DBA の結託

ユーザと DBA の結託によるデータへの不正アクセスである．

6.5.1　暗　　号

$sk_A = a, sk_A^{(D)} = a'^{-1}$ を得られるが，明らかに sk_l を求めることはできないので，データの機密性は保たれる．

6.5.2　検 索 タ グ

脅威 4 と同様に，任意の y と $\hat{y}$ の対応表を作ることができ，$\hat{y^*}$ から任意の平文 y^* を得ることができるが，同様の理由からもデータの機密性は守られる．

6.5.3　秘匿検索について

脅威 4 と同様に，x とプロキシが作成する落とし戸に含まれる $\hat{x}$ の対応表を作ることができるが，同じ理由から秘匿検索が可能である．

6.5.4　リボケーションについて

プロキシの再暗号化鍵がなければいかなるデータにもアクセスできないので，プロキシの再暗号化鍵を破棄するだけでリボケーション可能である．

6.6　6. 節のまとめ

五つの脅威を想定し，提案手法ではいずれの脅威に対してもデータの機密性が守られ，秘匿検索が可能であり，結託ユーザであっても効率的なリボケーションが可能であることを考察した．機密性について先行研究 [4] と比較した表を図 13 に示す．

7.　お わ り に

7.1　ま　と　め

本研究では先行研究 [4] に残された，ユーザとプロキシまたは DBA の結託によってデータの機密性が失われる，プロキシまたは DBA と結託しているユーザのリボケーションができない，範囲クエリに対応できない，個人間の関係を利用したセキュアなアクセス制御ができないという課題を解決する手法として，検索タグの作成方法と暗号化方法を提案した．

また，提案手法のセキュリティに関する考察を行い，データの機密性が保たれる，秘匿検索が可能である，任意のユーザの効率的なリボケーションが可能であることを確認した．

7.2　今後の課題

7.2.1　個人間関係の更新による鍵の変更

個人間関係が更新されるごとに再暗号化鍵や落とし戸の変換鍵を変更しなければならない．個人間関係の更新を再暗号化鍵に反映させる研究 [10] と統合するべきである．

7.2.2　クエリの実現方法の検討

提案手法は特定のデータ形式によらないが，本研究では RDF を想定している．そのため，提案した検索タグと落とし戸による検索を実際の SPARQL クエリとして実現する方法を検討しなければならない．

その際には，データの蓄積の効率化と検索の効率化を目的とした以下のような改善の余地がある．

検索のみに使われるデータは暗号文を作る必要は無く，タグだけ作成すればよい．無駄な暗号文の作成を省くことでデータの蓄積の効率化につながる．また，大小比較を必要とするデータは限られており，完全一致が多数を占めると考えられる．そこで，大小比較を必要としない属性のデータは，本研究で提案したタグではなく，暗号学的ハッシュ関数 [11] をベースとしたタグを付加することで検索速度の向上が見込め，検索の効率化につながる．

7.2.3　実験による評価

セキュリティと性能はトレードオフである．クエリの実現方法を検討した後，実際に実装してデータの蓄積，検索の実行時間を測定し，先行研究 [4] と比較することによって性能を評価する必要がある．

文　　献

[1] 独立行政法人 情報処理推進機構．災害に対応する IT システム検討プロジェクトチーム活動報告．January 2013. Retrieved December 18, 2017, from: https://www.ipa.go.jp/les/000026397.pdf.

[2] 総務省関東総合通信局防災対策推進室. 災害時に活用できる情報伝達手段. October 2014. Retrieved December 18, 2017, from: http://www.soumu.go.jp/main_content/000361388.pdf

[3] M. Blaze, G. Bleumer, and M. Strauss. Divertible protocols and atomic proxy cryptography. In Proceedings of Eurocrypt ‘98. Vol. 1403. 127-144.

[4] 児玉快，横田治夫．データやユーザの効率的な追加・削除が可能な秘匿情報アクセス手法．第 7 回データ工学と情報マネジメントに関するフォーラム，2015.

[5] Mark Giereth. On partial encryption of RDF-graphs. In Proceedings of the 4th International Conference on The Semantic Web, pp. 308-322. Springer-Verlag, 2005.

[6] G. Ateniese, K. Fu, M. Green, S. Hohenberger. Improved Proxy Re-Encryption Schemes with Applications to Secure Distributed Storage. In NDSS, pages 29-43, 2005.

[7] K. Lewi and D. J. Wu. Order-revealing encryption: New constructions, applications, and lower bounds. In ACM CCS, 2016.

[8] O. Goldreich, S. Goldwasser, S. Micali. How to construct random functions. J. ACM, 1986.

[9] R. A. Popa, C. M. S. Redfield, N. Zoldovich, and H. Balakrishnan. CryptDB: Protecting confidentiality with encrypted query processing. In Proc. of the 23rd SOSP, pages 85-100, Cascais, Portugal, Oct. 2011.

[10] 宮沢駿輔，平田拓三，Hieu Hanh Le，横田治夫．改竄への耐性と人との繋がりを表すグラフ更新の反映を可能とするアクセス権制御手法．第 10 回データ工学と情報マネジメントに関するフォーラム，2018.

[11] P. Rogaway and T. Shrimpton. Cryptographic Hash-Function Basics: Definitions, Implications, and Separations for Preimage Resistance, Second-Preimage Resistance, and Collision-Resistance. FSE 2004, LNCS 3017, 371-388.