暗号化データベース上の k-匿名化技術の提案

吉野 雅之[1] 鈴木 貴之[1] 長沼 健[1] 佐藤 尚宜[1]

概要：本稿では，一般化階層木を用い，暗号化されたデータベースを，暗号化したまま，k-匿名化する手法を提案する．一般化階層木は，予め準備してもよいが，本稿では，検索可能暗号で暗号化されたデータベースから，Huffman 符号等を用い，一般化階層木を生成する手法も併せて提案する．これにより，ユーザは一般化階層木を用意することなく，データベースの機密性を確保しながら，クラウド等の第三者機関へ匿名化処理を委託できる．また，委託される第三者機関にも，情報の覗き見に関する不正行為の可能性を排除できる等の利点がある．一般的な計算機上の実験では，100 万レコードの 3– 匿名化を 168 秒で遂行した．

k-Anonymization Technique over Encrypted Database

MASAYUKI YOSHINO[1] TAKAYUKI SUZUKI[1] KEN NAGANUMA[1] HISAYOSHI SATO[1]

1. はじめに

ウェアラブルデバイスや各種センサ機器の普及と共に，個人の移動情報や購買履歴情報が大量に収集され，その有用性に関する研究が，医療，広告，流通など様々な分野で行われている．日本では，2015 年に個人情報保護法が改正され，個人情報の利用目的として定めた目的以外での利用や，本人の同意なく第三者に提供することなどを可能にする，匿名加工情報が定義された．匿名加工情報では，自由な流通を実現可能とするため，産業を活性化する起爆剤として期待されている．

匿名加工情報の作成に用いる匿名化技術は，「個人の特定を困難にする技術」として，数多くの研究が報告されている．中でも，k-匿名性は個人の特定の難易度を定量化した代表的な指標として知られる [1]．k-匿名性とは，「テーブルの任意のレコードに対し，同じ値のレコードが自レコードを含め，k 個以上存在する」ことを示す．k-匿名性を満たす最適解を求めることは計算量困難な問題として知られ，その一部は NP Hard が証明されている [12]．従って，実用的には，最適解を得ることはあきらめ，多項式時間で動作可能な近似解を求める k-匿名化技術が用いられる．

他方，昨今の IoT 技術の進歩により，蓄積されるデータは大容量化が進み，単一システムでの管理は限界が近い．そこで，豊富な計算資源を有するクラウド等の外部システムと連携し，データ管理・処理の委託が進んでいる．しかしながら，k-匿名化処理の委託にあたっては，外部組織へ個人データへの流出につながる危険性から，普及が進んでいない．原理的には，委託前にデータを暗号化すれば情報漏洩は防止できるものの，匿名化処理の実行時には秘密鍵を渡す必要がある．結果，外部組織はデータを平文に戻してしまうため，依然として情報漏洩の問題は未解決である．このように，k-匿名化処理の委託は，個人データを対象とするため，難しいという課題があった．

プライバシ保護とデータ利活用を両立しながら，外部組織に処理を委託させるための研究も提案されている．例えば，Gentry らの方式 [13] や Ducas らの方式 [14] では，データを暗号化したまま，任意の演算処理が実行可能であるため，匿名化処理も安全に委託できる．しかしながら，これらの方式は暗号化状態での演算処理に要するオーバヘッドが未だ実用レベルからは遠く，小規模なデータベースの匿名化でさえ現実的ではない[*1]．

本稿は，この匿名化処理の委託と情報漏洩の防止を両立するため，暗号化したまま k-匿名化処理を実行する，暗号化プロトコルの定義とその構築法を提案する．提案方式を汎用 PC 上に実装し，実験を行ったところ，レコード数

[1] (株) 日立製作所 研究開発グループ 〒 244-0817 神奈川県横浜市戸塚区吉田町 292

[*1] 汎用的な計算機上で 1-NAND 演算を実行するため，Gentry らの方式は約 30 分，また高速化した Ducas らの方式でも約 1 秒を要する [13], [14]

100 万件のデータセットに対し，k-匿名化が 168 秒で実行可能なことを実証した．従って，実用的にも十分な高速性を達成可能との傍証を得た．

1.1 関連技術

(1)k-匿名化技術

個人識別の難易度を定量化する指標として，2002 年に Sweeney によって提案された k-匿名性がある [1]．k-匿名性を満たすためには，全ての属性が同一値であるレコードが k 件以上あるよう，レコードの値を変換する必要がある．この変換処理は再符号化と呼ばれ，さらに局所的再符号化方式と大局的再符号化方式に大別できる．局所的再符号化方式は，グループ化にレコード間の距離を計算するため，結果的に高い計算量が求められる．きめ細かい再符号化が行われるが，計算量の高さを考え，レコード数が少ないユースケースに利用が限定されがちである．一方，大局的再符号化方式は，その多くが一般化階層木と呼ばれる補助情報を使用し[*2]，距離の計算は行わず，規則的に再符号化を行う．高速性が長所であり，大規模なデータを対象とした k-匿名化に適する．

一般化階層木とは，レコードにおける同じ属性（列）同士のデータの関係を表す，木構造型の再符号化の指標である．元のデータを，葉ノードと呼び，最下位のノードとして位置付ける．また，最上位のノードを根ノードと呼ぶ．葉ノードから根ノードまでノード間で関係が定義される．下位のノードを一般化した値を上位のノードが保有する．例えば，最下位に葉ノードとして県単位{(北海道，東京)，(大阪，福岡)}，それぞれの上位ノードとしてより一般化された地域単位{東日本，西日本}，最上位の根ノードはもっとも一般化された国{日本}が位置する．

本稿では，大規模なデータを対象とするため，速度に優位性がある大局的再符号化方式を用いる．

(2) 検索可能暗号

検索可能暗号とは，古典的な暗号化機能に加え，暗号化したまま，2 種類の暗号化データの一致判定が可能な暗号技術である．暗号化や鍵を必要とするが，一致判定処理には，基本的には鍵を必要としない．具体的な検索可能暗号方式として [3], [4], [5], [6], [7] などが知られている．検索可能暗号は，共通鍵暗号方式と公開鍵暗号方式に分類される．共通鍵暗号方式は，暗号化用の鍵の公開はできないため，用途が限定されるが，高速処理に特徴があり，大容量データにも対応しやすい．一方，公開鍵暗号方式は，暗号化鍵を公開できるが，共通鍵暗号方式と比べ，処理性能が低い．本稿では，大規模なデータを対象としており，高速処理に優位な共通鍵暗号方式を採用する．

[*2] 一般化階層木を用いない k-匿名化技術もある [9]

1.2 本論文の貢献

データ分析のアウトソースとプライバシ保護を両立する，秘匿 k-匿名化技術を提案し，提案方式を汎用 PC 上に実装し，処理時間を計測し有用性検証を行った．レコード数 100 万件のデータセットに対し，k-匿名化 168 秒程度で実行可能なことを実証した．

2. 準備

2.1 用語

正の整数 m に対し，m 以下の正の整数の集合 $\{1, \cdots m\}$ を $[1, m]$ と表す．同様に，正の整数 n に対し，n 以下の正の整数の集合 $\{1, \cdots n\}$ を $[1, n]$ と表す．

2.2 テーブルの記法

まず，本稿で k-匿名化の対象とするデータベース内のテーブル TB を以下に定義する．

- テーブル TB を $(\mathbf{A}, \mathbf{R})$ の組とし，$\mathbf{A}$ は m 個の属性 A_i の配列 $(A_1, \cdots, A_m)$，$\mathbf{R}$ は n 個のレコード R_i の配列 $(R_1, \cdots, R_n)$ とする．レコード R_i は m 個のデータの配列 $(D_{i,1}, \cdots, D_{i,m})$ で構成される．また，データ $(D_{1,j}, \cdots, D_{n,j})$ をカラム C_j と呼ぶ．

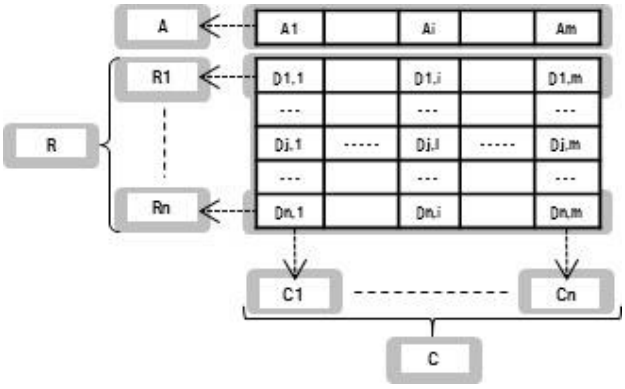

図 1 テーブル TB の構成

カラム C_j におけるデータ D の取りうる値域を平文空間 M_{C_j} とする．また，カラムの集合 $\mathbf{C}$ に対する平文空間を $\mathbf{M_C} = \sum_{j=1}^{m} M_{C_j}$ とする．

カラムの集合 $\mathbf{C}$ に対し，$\mathbf{C}(tk)$ をワード tk と等しいカラムの集合とする．また，ワードの集合 $\mathbf{tk} = \{tk_1, \ldots, tk_q\}$ と等しいカラムの集合を $\mathbf{C}(\mathbf{tk}) = \{\mathbf{C}(tk_1), \ldots \mathbf{C}(tk_q)\}$ とする．

次に，テーブル TB を暗号化したテーブル ETB を説明する．

- 暗号化テーブル ETB 内の用語は，テーブル TB の用語の頭文字に「E」を付けて表す．すなわち，暗号化テーブル ETB は $(\mathbf{EA}, \mathbf{ER})$ の組とし，$\mathbf{EA}$ は m 個の暗

号化した属性 EA_i の配列 $(EA_1, \cdots, EA_m)$，**ER** は n 個の暗号化したレコード ER_i の配列 $(ER_1, \cdots, ER_n)$ とする．暗号化レコード ER_i は m 個の暗号化データの配列 $(ED_{i,1}, \cdots, ED_{i,m})$ で構成される．また，暗号化したデータ $(ED_{1,j}, \cdots, ED_{m,j})$ を暗号化カラム EC_j と呼ぶ．

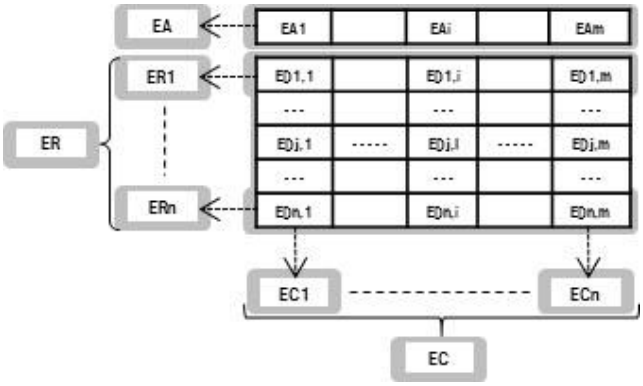

図 2 暗号化テーブル ETB の構成

最後に，テーブル TB の属性 **A** は全て k-匿名化の対象属性 (準識別子) とし，k-匿名化したテーブルを $kv-TB$，暗号化テーブル ETB を k-匿名化したテーブルを $kv-ETB$ とする．

2.3 暗号プリミティブ

本稿では，暗号プリミティブとして，共通鍵暗号 SKE(Symmtric Key Encryption) と共通鍵検索可能暗号 SSE(Searchable Symmetric Encryption) を用いる．

共通鍵暗号 SKE は初期設定 Setup, 暗号化 Enc, 復号化 Dec の 3 種類のアルゴリズムで構成される：

$$\mathtt{SKE} = (\mathtt{Setup}, \mathtt{Enc}, \mathtt{Dec})$$

共通鍵検索可能暗号 SSE は初期設定 Setup, 暗号化 Enc, トラップドア生成 Trpdr, 比較 Cmpr の 4 種類のアルゴリズムで構成される：

$$\mathtt{SSE} = (\mathtt{Setup}, \mathtt{Enc}, \mathtt{Trpdr}, \mathtt{Cmpr})$$

本稿が対象にする SSE は，暗号化データの検索に特化し，復号化 Dec は備えない．次に，SSE の各アルゴリズムの概要を説明する．

- Setup は，セキュリティパラメータ λ を入力し，秘密鍵 sk と公開パラメータ pp を出力する，確率的アルゴリズムである：$(sk, pp) \leftarrow \mathtt{Setup}(1^\lambda)$
- Enc は，秘密鍵 sk とデータ d を入力し，暗号文 ED を出力する，確率的アルゴリズムである：$ED \leftarrow \mathtt{Enc}(sk, D)$
- Trpdr は，秘密鍵 sk とデータ W を入力し，トラップドア TW を出力する，確定的アルゴリズムである：$TW \leftarrow \mathtt{Trpdr}(sk, W)$.
- Cmpr は，暗号化データ ED とトラップドア TW を入力し，$D = W$ か $D \neq W$ を (圧倒的な確率で) 判定する確定的アルゴリズムである：
  - If $D = W$, then $\mathtt{Cmpr}(ED, TD) = 1$.
  - If $D \neq W$, then $\mathtt{Cmpr}(ED, TD) = 0$ with probability $1 - \epsilon(\lambda)$ and ϵ is a negligible function.

3. 既存の k-匿名化手法

3.1 Greedy Search

k-匿名化手法は，一般化階層木を用いて対象データを匿名化する手法（大域的再符号化）と，それなしで匿名化する手法（局所的再符号化）に大別できる．匿名化データの用途が明確な場合には，匿名化の方針を明確化できる一般化階層木を用いる方が有利になりやすい．Wang らは，この一般化階層木に従いながら，逐次的に最適な匿名化対象のデータを選択する，貪欲法 (Greedy Search) を提案した [19]．このきめこまやかな匿名化処理により，大雑把な匿名化処理を施す，大域的再符号化の代表格である LeFerve らの手法 [2] よりも高品質な匿名化データを出力することが期待できる．

アルゴリズム 1: 大域的再符号化手法による k-匿名化の貪欲法 [19]

入力: 匿名化対象テーブル TB
出力: k-匿名化テーブル $kv-TB$

(1) TB から一般化階層木 **TR** を生成する (後述)
(2) **(テーブル TB が k-匿名性を満たすまで繰り返し)**
  (a) **TR** において（最近傍の親ノードへの）一般化により損失する情報エントロピーが最小の葉ノードを選択する
  (b) (a) の選択に従いテーブル TB を一般化する
  (c) (a) で選択した **TR** の葉ノードを削除し，その親ノードを **TR** の新たな葉ノードとする．
(3) TB を出力する

他方，匿名化処理をするために，匿名化対象のテーブルだけでなく，一般化階層木を事前に準備する必要がある．しかしながら，利用目的ごとに異なるデータが望まれるように，利用目的に合わせた匿名化処理をするには，一般化階層木を都度作成する必要がある．この一般化階層木の作成にかかる負荷を軽減するため，匿名化対象のテーブルから，一般化階層木を作成する手法が Harada らにより提案されている [10]．

Harada らの手法では，順序関係をもたない文字列等の属性用の一般化階層木は Huffman 符号を用いて作成する [18]．Huffman 符号はデータ圧縮に用いる符号木として

アルゴリズム 2: 一般化階層木の生成手法 [10]

入力: 匿名化対象テーブル TB
出力: 一般化階層木 $\mathbf{TR} = (TR_1, \ldots TR_m)$

(1) 全ての属性値の頻度を TB から数え上げ
(2) 全ての属性値を頻度の昇順にソートし，(属性値と頻度) をリスト Q の要素として保存する
(3) ソート後の属性値を一般化階層木 **TR** の葉ノードとする
(4) **(リスト Q の要素が 1 個になるまで繰り返し)**
  (a) 頻度最小のノード $2(a_1, a_2)$ 個を取り出し，リスト Q から削除する．
  (b) 新たに (a_1, a_2) の親ノード b を作成する．また親ノード b の頻度は子ノード (a_1, a_2) の頻度の和とする．
  (c) 親ノード b をリスト Q に追加し，頻度の昇順にソートする
(5) リスト Q の要素が 1 個である場合，対応する属性値を一般化階層木 **TR** の根ノードする
(6) 一般化階層木 **TR** を出力する

提案されたもので，あらゆる符号木の中で，符号木の高さの期待値を最小にすることが知られている．そのため，アルゴリズム 2 を用いて生成した一般化階層木は，Huffman 符号木により，頻度の低い属性値には低い階層が割り当てられ，結果として，過度の一般化を抑制する効果が期待できる．

3.2 k-匿名化処理を委託する際の課題

データ保有者がデータの k-匿名化処理を外部組織へ委託する場合を考える．委託には，外部組織へデータを開示する必要があるため，委託先からの情報漏洩のリスクが発生する．委託前にデータを暗号化すれば情報漏洩は原理的に防止できるが，この場合，秘密鍵を渡さなければ k-匿名化処理が代行できない．結果，外部組織に秘密鍵を渡す必要が生じ，問題は解決しない．

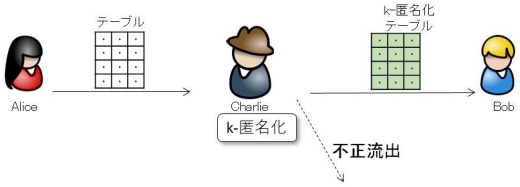

図 3 k-匿名化の委託における情報漏洩の危険性

4. 提案方式：秘匿 k-匿名化手法

4.1 戦略

確率的暗号は，高い安全性を有する反面，原理的に一切のデータ処理を委託できない，という利便性の課題がある．そこで，確率的暗号によりデータベースの安全性を確保し，k-匿名化処理を委託する際には暗号化データベースから確定値を委託先に開示する手法を採用する．なお，平文の一部をそのまま確定値として開示するのは情報漏洩に直結する危険性があるため，提案手法では確定値の導出に確定的暗号を用いる．

4.2 暗号化技術を用いた k-匿名化の処理フロー

暗号化技術の活用による k-匿名化処理の安全な委託処理フローの一例を図 4 に示す．

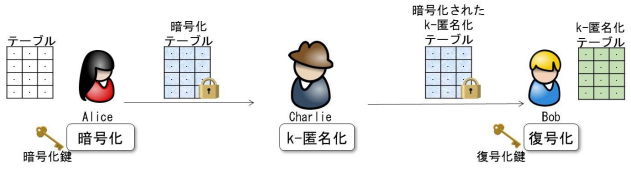

図 4 暗号化技術を用いた k-匿名化の処理フロー

図 4 の処理手順を示す．Alice は k-匿名化対象のテーブルを保有する．Charlie は k-匿名化処理の委託先，Bob は k-匿名化したテーブルの配送先である．Alice と Bob は事前に秘密鍵を共有しするが，Charlie は秘密鍵をもたないとする．

まず，Alice は，テーブルを暗号化し，この暗号化テーブルを Charlie に預ける．次に，Alice は k 匿名化処理をの依頼文（これも暗号化する）を Charlie に送る．暗号化したまま暗号化テーブルの k-匿名化処理を実行するため，秘密鍵を持たない Charlie にはテーブルの平文情報は漏洩しない．Charlie は作成した暗号化状態の k-匿名化テーブルを Bob へ送る．Bob は，これを復号し，k-匿名化テーブルを得る．この結果，Alice は，暗号化により情報漏洩を防止しながら，Charlie に k-匿名化処理を委託できる．

以下，暗号化関数として検索可能暗号を用い，この k-匿名化処理の実現方法を述べる．

4.3 暗号化 k-匿名方式 (EAS)

暗号化したまま，k-匿名化を実現する方式として，暗号化 k-匿名化方式 EAS(Encrypted k-Anonymization Scheme) を次のように定義する．

定義 1 *EAS* とは，以下の 5 つのアルゴリズムの組 (`Gen`, `Enc`, `Trpdr`, `Annmz`, `Dec`) で構成される．なお，平文空間などの公開パラメータは特にアルゴリズムの引数に記載しない．

- $ek, dk, tk \leftarrow \mathtt{Gen}(1^\lambda)$ は，暗号化鍵 ek, 復号化鍵 dk, トラップドア生成鍵 tk を生成する確率的アルゴリズムである．ただし，λ はセキュリティパラメータである．
- $ETB \leftarrow \mathtt{Enc}(ek, TB)$ は，暗号化テーブル ETB を生成する確率的アルゴリズムである．
- $\mathbf{td} \leftarrow \mathtt{Trpdr}(tk, \mathbf{A}, \mathbf{M_C})$ は，トラップドア **td** を生成

する確率的または確定的アルゴリズムである．

- $kv-ETB \leftarrow \mathtt{Annmz}(ETB, \mathbf{td}, kv)$ は，k-匿名化した暗号化テーブル $kv-ETB$ を生成する確定的アルゴリズムである．
- $kv-TB \leftarrow \mathtt{Dec}(dk, kv-ETB)$ は，k-匿名化したテーブル $kv-TB$ を返す確定的アルゴリズムである．

EAS が正しく動作するとは，任意の $\lambda \in \mathbb{N}$，テーブル TB と暗号化テーブル $ETB \leftarrow \mathtt{Enc}(ek, TB)$ に対し

$$kv-TB = \mathtt{Dec}(dk, \mathtt{Annmz}(ETB, \mathtt{Trpdr}(tk, \mathbf{A}, \mathbf{M_C}), kv))$$

が成立する場合を言う．

以下では，Alice が Charlie に k-匿名化を依頼し，Charlie から Bob に k-匿名化データを提供する際の，EAS プロトコルを紹介する．

EAS プロトコル

(0) 初期設定フェーズ

Alice は $\mathtt{Gen}(1^\lambda)$ を用い，暗号化鍵 ek，トラップドア生成鍵 tk を保有し，復号化鍵 dk を Bob に渡す．

(1) 登録フェーズ

Alice はテーブル TB を $\mathtt{Enc}(ek, TB)$ で暗号化し，得た暗号化テーブル ETB を Charlie に渡す．

(2) 依頼フェーズ

Alice は $\mathtt{Trpdr}(\mathtt{tk}, \mathbf{A}, \mathbf{M_C})$ でトラップドア $\mathbf{td}$ を生成し，k 値と共に Charlie に渡す．

(3) 匿名化フェーズ

Charlie は $\mathtt{Annmz}(ETB, \mathbf{td}, kv)$ を用い，k-匿名化した暗号化テーブル $kv-ETB$ を生成する．

(4) 提供フェーズ

Charlie は k-匿名化した暗号化テーブル $kv-ETB$ を Bob に渡す．Bob は $\mathtt{Dec}(dk, kv-ETB)$ を用い，k-匿名化したテーブル $kv-TB$ を得る．

EAS プロトコルでは，匿名化処理を委託された Charlie は秘密鍵を保持しないため，平文は開示されない点に注意する．Charlie 自身はもちろん，Charlie から第三者に暗号化テーブルが漏洩した際にもデータの機密性が確保できる．

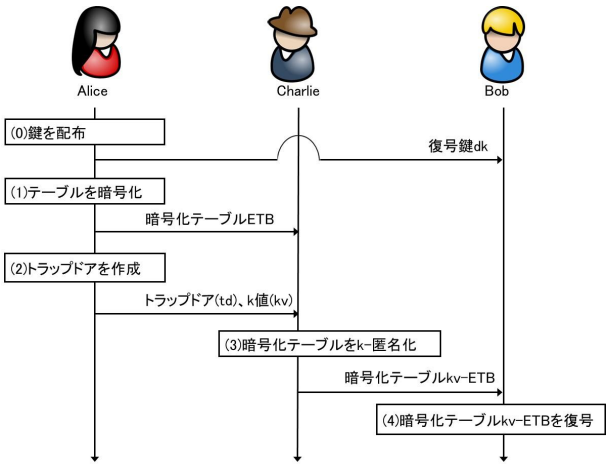

図 5 EAS プロトコルの処理手順

4.4 安全性

提案手法における，Charlie に漏洩する情報を述べる．

(A) 暗号化テーブル ETB から漏洩する情報を $L_1(ETB)$ とし，以下に示す．
・$L_1(ETB) = (\max\{|A_j|\}, \max\{|D_{i,j}|\}, m, n)$ ただし $i \in [1,n]$, $j \in [1,m]$.

(B) トラップドア $\mathbf{td}$ からの漏洩情報を $L_2(\mathbf{td})$ とし，以下に示す．
・$L_2(\mathbf{td}) = (\max\{|A_j|\}, \max\{|D_{i,j}|\}, m, |\mathbf{M_C}|)$
ただし $i \in [1,n]$, $j \in [1,m]$.

$L_1(ETB)$ の漏洩情報に加え，さらに $L_2(\mathbf{td})$ により新たに $|\mathbf{M_C}|$ が漏洩することに相当する．

(C) トラップドア $\mathbf{td}$ で ETB を検索すると，$\mathbf{td}$ と ETB の照合結果が分かる．この漏洩情報を $L_3(\mathbf{td}, ETB)$ とする．なお，$L_3(\mathbf{td}, ETB) = \mathbf{C}(\mathbf{tk})$ が成り立つ．

上記の漏えい関数 L_1，L_2，L_3 を用いて，EAS に対するシミュレーションベースの安全性 *Semantic Security under the Chosen Keyword Attack* ($EAS-SS-CKA$) を次のように定義する．

定義 2 ***EAS-SS-CKA* 安全**: $\mathcal{A}$ と $\mathcal{S}$ をステートフルな PPT アルゴリズムとする．このとき，以下のような 2 つの確率的な試行 $Real_{\mathcal{A}}$, $Ideal_{\mathcal{A},\mathcal{S}}$ を考える：

$Real_{\mathcal{A}}$: $\mathcal{C}$ は $ek, dk, tk \leftarrow \mathtt{Gen}(1^\lambda)$ を動作させる．$\mathcal{A}$ はテーブル TB を $\mathcal{C}$ に渡す．$\mathcal{C}$ は $ETB \leftarrow \mathtt{Enc}(ek, TB)$ より暗号化テーブル ETB を得る．$\mathcal{C}$ は $\mathbf{td} \leftarrow \mathtt{Trpdr}(tk, \mathbf{A}, \mathbf{M_C})$ よりトラップドア $\mathbf{td}$ を得る．$\mathcal{C}$ は $\mathcal{A}$ に ETB と $\mathbf{td}$ を渡す．$\mathcal{A}$ は $\mathtt{b} \in \{0,1\}$ を出力する．

$Ideal_{\mathcal{A},\mathcal{S}}$： 出題者 $\mathcal{C}$ は，テーブル TB を $\mathcal{S}$ に渡す．$\mathcal{S}$ は $L_1(TB)$ に基づき，ETB を生成し，$\mathcal{A}$ に渡す．その後 $\mathcal{A}$ は，$L_2(\mathbf{A}, \mathbf{M_C})$ に基づき，$\mathbf{td}$ を生成し，$\mathcal{A}$ に渡す．$\mathcal{A}$ は $\mathtt{b} \in \{0,1\}$ を出力する．

EAS が *EAS-SS-CKA* 安全であるとは，任意のセキュリティパラメータ λ，任意の PPT アルゴリズム $\mathcal{A}$ に対し

$$|Pr[Real_{\mathcal{A}}(\lambda) = 1] - Pr[Ideal_{\mathcal{A},\mathcal{S}}(\lambda) = 1] \leq negl(\lambda)$$

を満たすような PPT アルゴリズム $\mathcal{S}$ が存在することをいう．

匿名化に関するクエリにより，$\mathbf{C}(\mathbf{tk})$ が開示される．しかし，平文が漏えいするわけではなく，$\mathbf{td}$ から導出される確定値が開示されるだけであることに注意されたい．例えば，この確定値の導出に，鍵付きハッシュ関数を用いれば，未だ一定の安全性レベルが保証される．具体的には，すべての確定値が現れる状況は [21] の提案方式と同等の安全性を有する．

4.5 具体的な構成法

共通鍵暗号 SKE と検索可能暗号 SSE を用いた，EAS の具体的な構成法を示す．

$\mathtt{Gen}(1^\lambda)$: $sk_1 \leftarrow \mathtt{SSE.Gen}(1^\lambda)$，$sk_2 \leftarrow \mathtt{SKE.Gen}(1^\lambda)$ を生成し，暗号化鍵 ek を sk_1，トラップドア生成鍵 tk を (sk_1, sk_2)，復号化鍵 dk を sk_2 とする．

$\mathtt{Enc}(ek, TB)$: 値によらず長さが同じになるよう，属性，データをパディングする．

- $A_j \leftarrow A_j \| 00\ldots0$ ただし $|A_j| = \max\{|A_i|\}$, $i \in [1,m]$.
- $D_{i,j} \leftarrow D_{i,j} \| 00\ldots0$ ただし $|D_{i,j}| = \max\{|D_{i,k}|\}$, $i \in [1,m]$, $j,k \in [1,n]$.

次に，パディングした属性とデータをそれぞれ SSE で暗号化する．

- $\{EA_j \leftarrow \mathtt{SSE.Enc}(sk_1, A_j)\}$ ただし $j \in [1,m]$
- $\{EC_{i,j} \leftarrow \mathtt{SSE.Enc}(sk_1, C_{i,j})\}$ ただし $i \in [1,m]$, $j \in [1,n]$

$\mathtt{Trpdr}(tk, \mathbf{A}, \mathbf{M_C})$: 属性とデータ向けにトラップドアを SSE で作成する．また，k-匿名化の一般化処理の暗号化データを SKE で作成する．

- $\{TA_j \leftarrow \mathtt{SSE.Trpdr}(sk_1, A_j)\}$ ただし $j \in [1,m]$
- $\{TQ_{i,j} \leftarrow \mathtt{SSE.Trpdr}(sk_1, x_{i,j})\}$ ただし $i \in \mathbb{Z}^+_{|M_{C_j}|}$, $j \in [1,m]$
- $\{EB_j \leftarrow \mathtt{SKE.Enc}(sk_2, A_j)\}$ ただし $j \in [1,m]$
- $\{ED_{i,j} \leftarrow \mathtt{SKE.Enc}(sk_2, x_{i,j})\}$ ただし $x_{i,j} \in M_{C_j}$, $i \in \mathbb{Z}^+_{|M_{C_j}|}, j \in [1,n]$

$j \in [1,n]$ をランダム置換し (以降，$j \to h$ とする)，$(\{TA_h\}\{EB_h\}\{TQ_{i,h}\}\{ED_{i,h}\})$ を $\mathbf{td}$ として出力する．

$\mathtt{Annmz}(\mathbf{td}, TB, kv)$: SSE の Cmpr を用い，$k$-匿名化を行う．

1. 全ての $j \in [1,n]$ で，$\mathtt{SSE.Cmpr}(EA_j, TA_h) = 1$ を満たす (j,h) の組み合わせを見つける．
2. 暗号化された属性 EA_h ごとに，トラップドアと一致する暗号化データ $EC_{i,j}$ を集計する (集計値は sum_{EB_h} とする)．

$$sum_{EB_h} = \sum_{i'=1}^{M_{C_j}} \sum_{i=1}^{n} \mathtt{SSE.Cmpr}(EC_{i,j}, TQ_{i',h})$$

3. 2. の集計値 sum_{EB_h} を暗号化された属性 EB_h の頻度とする．[10] に従い，暗号化された属性 EB_h ごとに一般化階層木を作成する[*3]．このとき，一般化階層木のノードのラベルを $R_{i',h}$ の値とする．
4. 一般化階層木を用い，k-匿名性を満たすまで，レコードの一般化を繰り返す．
5. 4. の結果に基づき，一般化階層木のノードのラベルを暗号化テーブル ETB におけるデータと置換し，$ETB = (\mathbf{EB}, \mathbf{ED})$ を作成する．ただし，$\mathbf{EB} = (EB_1, \cdots, EB_n)$， $\mathbf{ED} = (ER_1, \cdots, ER_n)$ である．
6. $\mathbf{ED}$ のレコード $(ER_1, \cdots, ER_n)$ の順番をランダム置換する．
7. k匿名化された暗号化データ $kv - ETB$ として $(\mathbf{EB}, \mathbf{ED})$ を出力する．

$\mathtt{Dec}(dk, ETB)$： 暗号化テーブルの暗号化された属性と暗号化データを SKE で復号化する．

- $\{A_j \leftarrow \mathtt{SKE.Dec}(sk_2, EB_j)\}$ ただし $j \in [1,n]$
- $\{C_{i,j} \leftarrow \mathtt{SKE.Dec}(sk_2, ED_{i,j})\}$ ただし $i \in [1,m]$, $j \in [1,n]$

5. 実験による性能評価

5.1 性能見積

一般化階層木の生成では，レコードの出現頻度の数え上げと，出現頻度のソートが主要な処理である．レコード数は n であるので，一つの属性におけるソートの処理時間は $O(n \log n)$ である．属性数 m であることから，一般化階層木の生成時間は $O(mn \log n)$ と評価できる．一般化階層の

[*3] 集計結果に基づき，出現頻度の少ないノードから徐々に一般化階層木の上位ノードを形成する．上位ノードの出現頻度は下位ノードの出現頻度の総和であり，一般化の度合いが強すぎないよう，Huffman 符号等を用いる．また，下位ノードを一般化した上位ノードは下位ノードの値の論理和 (or) で表現される．

生成とは異なり，匿名化プロセスは平文データベースと暗号化データベースで比較処理を除き, 等しい.

5.2 実験評価

想定される評価用データは，日本人の統計情報を参考に([17])，10^3 レコードから 10^6 レコードを段階的に用意した．レコード規模ごとに実装性能を測定するため，生成した模擬データを表 1 に示す.

表 1 匿名化対象データベースの属性とその値

属性値	取りうる値
職種	[1, 24] のいずれかの整数値
性別	男性 or 女性
住所	5000 種類の住所とランダムな丁目，番地の掛合せ
生年月日	西暦表記の数字 8 桁 (yyyymmdd)

性能検証に用いた計算機のスペックを表 2 に示す.

表 2 評価用の計算機環境

CPU	Core i7 6700@3.4GHz
RAM	32GB
OS	CentOS 7.4
JVM	1.8.0

実装では，SKE は 256-bit AES を SSE は Yoshino らの方式 [7] を採用した．表 3 に，暗号化されたデータの頻度を集計後，Huffman 符号木に基づく一般化階層木の生成にかかる時間を示す[*4].

表 3 一般化階層木の生成時間 (4 属性分)

レコード数	平文	暗号文
1,000 件	83.5 ミリ秒	86.0 ミリ秒
10,000 件	185.9 ミリ秒	210.3 ミリ秒
100,000 件	361.7 ミリ秒	394.7 ミリ秒
1,000,000 件	675.2 ミリ秒	845.6 ミリ秒
10,000,000 件	1709.6 ミリ秒	2143.2 ミリ秒

年齢等の数字データを扱う場合，厳密には，一般化階層木の生成には，3.1 章で述べた Huffman 符号木の代わりに，順序関係を考慮した Hu-Tucker 符号木を用いる必要がある．Hu-Tucker 符号木も，Huffman 符号木と同様，一般化階層木の深さの期待値を最小にすることが知られている.

表 4 は，データ読込，一般化階層木の生成を完了した状態から，3-匿名化データの作成処理に係る時間を示す.

表 4 に示すように，暗号化状態での k-匿名化処理は，平文と比べ遜色ない．1,000,000 件まではレコード数が増えるにつれ処理時間が増加した。1,000,000 件では，逆にデータ数の多さから k-匿名性を満たしやすい状況が生まれ，処理時間は短縮されている.

[*4] Annmz の Step.3 の処理を対象とする．平文データベースの読込時間は含まない．また，Step. 1 と Step.2 は前処理として，終了した状態で実施する.

表 4 k-匿名化処理時間の評価 ($k = 3$)

レコード数	平文	暗号文
1,000 件	434 ミリ秒	440 ミリ秒
10,000 件	6,006 ミリ秒	5,977 ミリ秒
100,000 件	71,086 ミリ秒	79,594 ミリ秒
1,000,000 件	150,111 ミリ秒	168,447 ミリ秒
10,000,000 件	30,194 ミリ秒	36,605 ミリ秒

6. まとめ

本稿では，匿名化処理の安全な委託を実現するため，暗号化データベース上で k-匿名化を実現する手法を提案した．提案手法は，暗号化されたデータベースから，k-匿名化用の一般化階層木を生成できる．また，一般化階層木を与えた場合，汎用的な計算機上で 100 万レコードの k-匿名化を 168 秒で処理できることを確認した．暗号化により，課題であった k-匿名化の処理委託先への情報漏洩を防止でき，さらに委託先にも意図しないデータの覗き見を防止できる効果がある.

参考文献

[1] Latanya Sweeney. k-Anonymity: A Model for Protecting Privacy. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems 10(5), pages 557–570, 2002.

[2] Kristen LeFevre, David J. DeWitt, Raghu Ramakrishnan. Incognito: Efficient Full-Domain K-Anonymity. SIGMOD Conference 2005: pages 49–60, 2005.

[3] Dan Boneh, Giovanni Di Crescenzo, Rafail Ostrovsky, and Giuseppe Persiano. Public key encryption with keyword search. In *EUROCRYPT*, volume 3027 of *Lecture Notes in Computer Science*, pages 506–522. Springer, 2004.

[4] Dan Boneh and Brent Waters. Conjunctive, subset, and range queries on encrypted data. In Salil P. Vadhan, editor, *TCC*, volume 4392 of *Lecture Notes in Computer Science*, pages 535–554. Springer, 2007.

[5] Reza Curtmola, Juan A. Garay, Seny Kamara, and Rafail Ostrovsky. Searchable symmetric encryption: improved definitions and efficient constructions. In *ACM Conference on Computer and Communications Security*, pages 79–88. ACM, 2006.

[6] Arian Perrig Dawn Xiaodong Song, David Wagner. Practical techniques for searches on encrypted data. In *the 2000 IEEE Symposium on Security and Privacy*, pages 44–55, 2000.

[7] Masayuki Yoshino, Ken Naganuma, Hisayoshi Satoh. Symmetric Searchable Encryption for Database Applications. NBiS 2011: 657-662

[8] 長沼健，佐藤尚宜，吉野雅之，佐藤嘉則，検索可能暗号を用いた秘匿分析手法，SCIS2014 予稿集

[9] Kristen LeFevre, David J. DeWitt, Raghu Ramakrishnan. Mondrian Multidimensional K-Anonymity. ICDE 2006

[10] Kunihiko Harada, Yoshinori Sato, Yumiko Togashi. Reducing Amount of Information Loss in k-Anonymization for Secondary Use of Collected Per-

sonal Information. SRII Global Conference 2012: 61-69
[11] Dan Boneh, Kevin Lewi, Hart William Montgomery, Ananth Raghunathan. Key Homomorphic PRFs and Their Applications. IACR Cryptology ePrint Archive 2015: 220 (2015)
[12] Adam Meyerson, Ryan Williams: On the Complexity of Optimal K-Anonymity. PODS 2004: 223-228
[13] C. Gentry. Fully homomorphic encryption using ideal lattices. In 41st ACM STOC, pages 169-178, 2009.
[14] L. Ducas and D. Micciancio. FHEW: Bootstrapping homomorphic encryption in less than a second. In Eurocrypt, pages 617-640, 2015.
[15] 竹之内隆夫，側高幸治，豊田由起，高橋翼，森拓也．部分データセットとの突合に対する耐性を有するレセプト匿名化方式．医療情報学 33(3), 127-138, 2013.
[16] Kimura E, Chida K, Ikarashi D, Hamada K, Ishihara K., “Statistical disclosure limitation of health data based on Pk-anonymity”, Studies in health technology and informatics, 180:1117-9, August 2012.
[17] 総務省統計局. 人口推計-平成 29 年 7 月報-. http://www.stat.go.jp/data/jinsui/pdf/201707.pdf.
[18] Huffman, D.:A method for the construction of minimum-redundancy codes, Institute of Radio Engineers, Vol.40, No.9, pp.1098-1102 (1952).
[19] K. Wang, P. Yu, and S. Chakraborty, “Bottom-up generalization: A data mining solution to privacy protection”, in 4th IEEE Int. Conf. Data Mining, pp. 279–288, 2004.
[20] R. Curtmola, J. Garay, S Kamara, and R. Ostovsky. Searchable symmetric encryption: Improved definitions and efficient constructions. *Journal of Computer Security*, 19(5): 895–934, 2011.
[21] W. Ogata, K. Koiwa, A. Kanaoka, and S. Matsuo. Toward practical searchable symmetric encryption. In IWSEC 2013, volume 8231 of LNCS, pages 151-167, 2013.