Computer Security Symposium 2017
23 - 25 October 2017

新たな最適化問題による順序保存型暗号化データベースに対する推論攻撃

小野澤 綜大[1,a)]　國廣 昇[1]　吉野　雅之[2]　長沼 健[2]

概要：CryptDB は復号せずに暗号化データの検索ができる RDB である．ACM CCS2015 で Naveed らは CryptDB に対して Cumulative 攻撃を提案した．この攻撃は目的関数を定義して最適化することで暗号化データの復元を行う．論文中の実験では暗号化データが多い場合には攻撃が成功している．SCIS2017 で我々は，目的関数の変更により Cumulative 攻撃の改良を行っているが，成功率がそれほど高くないという問題があった．この論文では，最適化の際に，あらかじめ順序の制約を加える手法を提案する．数値実験を行い，提案手法は，従来の方式より高い成功率を示すことを確認した．

キーワード：順序保存型暗号，暗号化データベース，推論攻撃

Improved Inference Attack on Order-Preserving Encrypted Databases based on New Optimization Problem

Sota Onozawa[1,a)]　Noboru Kunihiro[1]　Yoshino Masayuki[2]　Ken Naganuma[2]

Abstract: CryptDB is a relational Database that allows an encrypted search. At ACM CCS2015, Naveed et al. proposed the cumulative attack on CryptDB. This attack recovers encrypted data by optimizing cost function In that paper, experimental results show that their attack succeeds to recover the plaintext when the attacker has a lot of encrypted data. At SCIS2017, we improve the cumulative attack by modifying the cost function. However, this attack does not gain high success rate In this paper, we propose a new attack by modifying the optimization problem to a new optimization problem with order restrictions. Finally, we give experimental results to show that our proposed attack is better than the cumulative attack.

Keywords: Order-Preserving-Encryption, Encrypted-Database, Inference attack

1. はじめに

個人的な情報を扱うサービスを運用する際，暗号化してデータを保存することはセキュリティ安全性の観点から重要である．しかし，近年，扱うデータの規模が大きくなりつつあり，暗号化は利便性の観点から見ると大きな障壁となる．一般的に用いられている暗号化方式を用いてデータを暗号化した場合，データを利用するたびに全てのデータを復号する必要があり，特に，大規模なデータを扱う場合，復号処理を伴うため非効率的である．

そこで注目されているのが暗号化データベースである．暗号化データベースとは暗号化したデータ上での検索が可能なデータベースである．暗号化したまま検索を行なった後に復号を行うことが可能なので，復号処理は必要なデータに限定して行うことができる．よって，暗号化したすべてのデータに対して復号を行う必要がなく，復号処理の負担を減らすことが可能となる．このような背景から暗号化データベースへ注目が集まっている．

暗号化データベースの実現方法として，Property-Preserving Encryption (PPE) を暗号化に用いる方法がある．PPE とは平文のある特徴を暗号文に残したまま暗号

[1] 東京大学大学院新領域創成科学研究科, 〒 277-8561 柏市柏の葉 5-1-5
[2] 日立製作所, 神奈川県横浜市戸塚区吉田町 292 番地
[a)] 5273985647@edu.k.u-tokyo.ac.jp

化する暗号化方式の総称である．例えば，平文の大小関係が暗号文においても成立する順序保存暗号 (OPE)[1], [2], [3] や決定的な暗号 (DTE) がある．また，検索可能暗号なども用いられる．

PPE を用いた暗号化データベースの代表例として，CryptDB [8] がある．CryptDB は商用のリレーショナルデータベースに劣らない検索機能と検索速度をもつ暗号化データベースである．CryptDB では DTE，OPE，検索可能暗号などが用いられている．DTE は一致検索を，OPE は数値検索を，検索可能暗号は文字検索を暗号化したデータ上で行うために用いられる．Google の Encrypted BigQuery は CryptDB をベースに開発されている．CryptDB は実社会への応用が期待されている暗号化データベースである．

Naveed らは PPE を用いた暗号化データベースに対する攻撃を提案した [9]．彼らの研究では暗号化されたデータベースに DTE が用いられている場合と，OPE が用いられている場合に対して攻撃を行っている．さらに，Naveed らは National Inpatient Sample(NIS) が公開している医療データのデータセットを，補助的な情報として用いた実験を行っている．この実験により，暗号化データが十分多い時にはデータの復元が可能であることを示した．

この論文では，Naveed らの研究で提案された Cumulative 攻撃の改良を行う．本研究では，新たに最適化問題を定義しその問題を解くことで暗号化データを復元する．

1.1 関連研究

数値検索を可能にする暗号化方式 OPE や Order-Revealing Encryption(ORE) [5], [6], [7] に対する，様々な解析が行われている．Lewi らは ORE から漏れる順序情報に基づき，2 次元データに対する攻撃を行なった [11]．また，Horst らは新たに提案された OPE[4] に対して，その構成方法に着目して攻撃を行なった [10]．

2. 従来手法 [9]

2.1 推論攻撃

平文空間を $\mathbb{M}$，暗号文空間を $\mathbb{C}$ とする．生データ $\boldsymbol{m} = (m_1, \ldots, m_k)$ に対応する暗号化データを $\boldsymbol{c} = (c_1, \ldots, c_k)$ とする．推論攻撃は，補助的なデータを必要とする攻撃である．補助的なデータとは，公開データであり，誰でも容易に入手できるデータである．この論文では，生データと補助的なデータは，各要素の出現頻度がほとんど一致していることを仮定している．補助的なデータを利用し，暗号化データ $\boldsymbol{c} = (c_1, \ldots, c_k)$ から生データ $\boldsymbol{m} = (m_1, \ldots, m_k)$ を復元する攻撃を総称して，推論攻撃と呼ぶ．

推論攻撃の中で最も基本的な攻撃が頻度解析である．暗号化データと補助的なデータを頻度順に対応させる攻撃である．CryptDB に対する攻撃に対しては，年齢や性別データを復元する事が想定される．その場合には，国勢調査などが補助的なデータの候補になる．[9] では，OPE を用いた暗号化データベースに対する攻撃として，Cumulative 攻撃を提案している．

2.2 ヒストグラム，累積分布関数

OPE を決定的な暗号とみなすことができる．すなわち，m_1=m_2 ならば，常に，$\mathrm{Enc}(m_1) = \mathrm{Enc}(m_2)$ である．そのため，暗号文から平文の頻度情報を得ることが可能である．また，OPE では順序情報を暗号文から得ることができるため，頻度情報と順序情報の両方を用いて推論攻撃を行うことができる．[9] では頻度を定量化するためにヒストグラムを用い，順序情報を定量化するために累積分布関数（CDF）を用いている．

データの集合 $\mathbb{D}$ は順序が定義されている集合とする．データ $\boldsymbol{d}$ を $\boldsymbol{d} = (d_1, \ldots, d_k)\,(d_i \in \mathbb{D}, 1 \leq i \leq k)$ と書くこととする．$\boldsymbol{d}$ のヒストグラムは $\mathbb{N}_{\geq 0}$ 上 $|\mathbb{D}|$ 次元ベクトルであり，$\mathbf{Hist}\,(\boldsymbol{d}) = (h_1, \ldots, h_n)$ を $\boldsymbol{d}$ のヒストグラムとする．ただし，h_i は i 番目に小さい要素の個数とする．例えば，$\mathbb{D} = \{1,2,3,4,5\}, \boldsymbol{d} = (1,1,2,3,5,1,3)$ とすると，$\mathbf{Hist}\,(\boldsymbol{d}) = (3,1,2,0,1)$ である．

次に $\boldsymbol{d}$ の累積分布関数 (CDF) を定義する．$f_i = \sum_{j=1}^{i} h_j$ とした上で，$\mathbf{CDF}\,(\boldsymbol{d}) = (f_1, \ldots, f_n)$ と定義する．例えば，$\mathbb{D} = \{1,2,3,4,5\}, \boldsymbol{d} = (1,1,2,3,5,1,3)$ とすると，$\mathbf{CDF}\,(\boldsymbol{d}) = (3,4,6,6,7)$ である．

2.3 Cumulative Attack [9]

前述したように，OPE に対しては，頻度と順序の両方の情報を用いて攻撃することが可能である．順序の情報を用いた最も単純な攻撃は，得られた暗号化データを順番に平文の順序に基づき当てはめる攻撃であり，ソート攻撃と呼ばれる．この攻撃は，全ての種類の暗号文が暗号化データとして得られた場合に成功する．その一方で，全ての暗号文が得られない場合，失敗するという問題点がある．

[9] では，順序情報だけでなく，頻度情報と組み合わせる考える攻撃が提案されている．全ての平文に対する暗号文が得られない場合でも，補助的なデータから得られるデータの情報を用いることで，平文を復元する攻撃として，Cumulative 攻撃 [9] が提案されている．

攻撃者は，暗号化データ $\boldsymbol{c}$，補助的なデータ $\boldsymbol{z}$ を得ている状況を考える．Cumulative 攻撃は次の目的関数を最小化する n 次置換行列 P を求めることにより復元結果を得る．

$$\sum_{i=1}^{|\mathbb{M}|} \left(|\mathbf{Hist}\,(\boldsymbol{c})_i - \langle \boldsymbol{P}_i, \mathbf{Hist}\,(\boldsymbol{z}) \rangle|^2 + |\mathbf{CDF}\,(\boldsymbol{c})_i - \langle \boldsymbol{P}_i, \mathbf{CDF}\,(\boldsymbol{z}) \rangle|^2 \right) \tag{1}$$

ここで，置換行列とは各行，各列に 1 が 1 つだけ存在して，他の成分は 0 となる行列である．$\boldsymbol{P}_i$ は行列 P の i 行目のベクトルであり，$\mathbf{Hist}\,(\boldsymbol{c})_i$ はヒストグラムの i 番目の成

分，$\langle \boldsymbol{P}_i, \mathbf{Hist}(\boldsymbol{z}) \rangle$ は $\boldsymbol{P}_i$ と $\mathbf{Hist}(\boldsymbol{z})$ の内積を表している．ヒストグラムは頻度情報の当てはまりの良さを，CDF は順序情報の当てはまりの良さを定量化している．

$\boldsymbol{P}_i$ の j 番目の成分が 1 のとき，目的関数の i 番目の項は

$$\left|\mathbf{Hist}(\boldsymbol{c})_i - \mathbf{Hist}(\boldsymbol{z})_j\right|^2 + \left|\mathbf{CDF}(\boldsymbol{c})_i - \mathbf{CDF}(\boldsymbol{z})_j\right|^2 \tag{2}$$

と変形することができる．これは，i 番目に小さい暗号文を j 番目に小さい平文に対応させた時の誤差を表している．すなわち，目的関数を最小にする置換行列 P を求めることができれば，誤差が最小となる暗号文と平文の対応関係が得られることになる．Cumulative 攻撃をまとめると次のようになる．ただし，$n = |\mathbb{C}|$ であり，$\mathbb{P}_n$ は n 次置換行列全体の集合である．

Algorithm 1 Cumulative Attack

1. ヒストグラムを計算する．

$\boldsymbol{\psi} \leftarrow \mathbf{Hist}(\boldsymbol{c}), \boldsymbol{\pi} \leftarrow \mathbf{Hist}(\boldsymbol{z})$

2. CDF を計算する．

$\boldsymbol{\varphi} \leftarrow \mathbf{CDF}(\boldsymbol{c}), \boldsymbol{\mu} \leftarrow \mathbf{CDF}(\boldsymbol{z})$

3. 目的関数を最小化する P を求める．

$P^* \leftarrow \underset{P \in \mathbb{P}_n}{\operatorname{argmin}} \sum_{i=1}^{|\mathbb{M}|} \left(|\psi_i - \langle \boldsymbol{P}_i, \boldsymbol{\pi} \rangle|^2 + |\varphi_i - \langle \boldsymbol{P}_i, \boldsymbol{\mu} \rangle|^2\right)$

[9] では，目的関数の最小化を Linear Sum Assignment Problem (LSAP) に帰着させ，ハンガリアンアルゴリズム [12], [13] を用いて解いている．ステップ 3 が計算時間としては支配的であり計算時間は $O(n^3)$ である．

LSAP は n 次元コスト行列 $C = (c_{ij}), (0 \leq c_{ij})$ に対して，次のように定式化される．

$$\begin{aligned}
&\text{minimize} \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}X_{ij} \\
&\text{subject to} \sum_{i=1}^{n} X_{ij} = 1, 1 \leq j \leq n \\
&\sum_{j=1}^{n} X_{ij} = 1, 1 \leq i \leq n \\
&X_{ij} \in \{0,1\}, 1 \leq i,j \leq n
\end{aligned}$$

Cumulative 攻撃はコスト行列 $C = (c_{ij})$ を

$$c_{ij} = \left|\mathbf{Hist}(\boldsymbol{c})_i - \mathbf{Hist}(\boldsymbol{z})_j\right|^2 + \left|\mathbf{CDF}(\boldsymbol{c})_i - \mathbf{CDF}(\boldsymbol{z})_j\right|^2 \tag{3}$$

と設定することにより，LSAP に帰着させている．ここで，c_{ij} は暗号文 c_i を平文 m_j に対応させたときの暗号文と平文のヒストグラム，累積分布関数の差の二乗和を表している．

2.4 Cumulative 攻撃の問題点

Cumulative 攻撃には以下の 2 つの問題点がある．

- 暗号化データの正確なヒストグラムが計算できない．
- 復元結果と順序が矛盾することがある．

以下，順に説明する．

まず前者の問題点を説明する．暗号化データ $\boldsymbol{c}$ に実際に現れる暗号文の集合を $\mathbb{C}'$ として，$|\mathbb{C}'| = l < |\mathbb{C}|$ が成り立つ状況を考える．ヒストグラムが定まるとコスト行列は一意に定まることに注意すると，ヒストグラムを適切に定めることができれば十分である．一方で，攻撃者は，$\mathbb{C}$ の要素を全て知る事はできない．そこで，仮にヒストグラムを次のように設定してみる．

$$\mathbf{Hist}(\boldsymbol{c}) = (h_1, \ldots, h_l, 0, \ldots, 0) \tag{4}$$

ただし，[9] では，明示的には書かれていないものの，0 で埋める位置は補助的なデータを見て，出現頻度の低い値の位置に 0 を埋めていると推測される．この例では，値が大きくなるほど出現頻度が低くなる補助的なデータが得られたとする．

このとき，暗号文の頻度情報を表す $\boldsymbol{\psi}$ に関しては一見、問題ないように見える．しかし，累積分布関数をこのヒストグラムに対して計算した場合 $\mathbb{C} \setminus \mathbb{C}'$ の元がすべて後方にあるという情報を元に復元してしまう．暗号データに現れていない暗号文は補助的なデータの中で出現頻度が低い平文に対応するとは限らないので，誤った情報を元に復元する場合がある．また，攻撃者は $\mathbb{C} \setminus \mathbb{C}'$ の元が $\mathbb{C}$ の中で何番目に大きい元なのかを知らないため，正しい位置に 0 を入れたヒストグラムを計算する事はできない．また，埋め方は，$\binom{|\mathbb{C}|}{|\mathbb{C}'|}$| 通りあるので，すべて計算して目的関数が最小となる埋め方を選択する方法は効率が悪く，良いヒストグラムの 0 の埋め方を探すのが困難である．

次に後者の問題点を説明する．$c_i < c_j$ となる暗号文にたいして $c_i \to m_i$, $c_j \to m_j$ で $m_j < m_i$ となる復元結果が Cumulative 攻撃によって得られたとする．この復号結果は，暗号文から得られる平文の順序と復元した平文の順序とで整合性が取れていない．また，アルゴリズムの特性上，このような不整合が生じることを禁じていない．

以上の二つの問題点を克服し，常に正しい順序になる復号結果を出力するアルゴリズムがの提案を目標とする．

3. 順序保存割り当て問題の導入と提案アルゴリズム

Cumulative 攻撃には 2 つの問題があったが，LSAP を新たな最適化問題に変更することでこれらの問題を解決する．新たな最適化問題は，LSAP を改良し，順序の制約を新たに設けることにする．これにより，順序が崩れる問題を克服する．さらに，新たな最適化問題の解法アルゴリズムの入力は正方行列である必要がなくなるため，暗号化データのヒストグラムを 0 で埋めるという不自然な処理が不要となる．これにより，暗号化データの正確なヒストグラムが計算できない問題を解決することができる．

既存手法ではLSAPに帰着させ，式 (1) を最小とする P を求めていた．その一方で，我々の提案手法は順序の制約を持つ最適化問題に帰着させ，式 (1) を最小とする P を求める．

3.1 新たな最適化問題

次の式 (5)(6) によって順序の制約を定義する．

$$X_{ij} = 1 \text{ ならば } X_{i'j'} \neq 1 \, (i < i', j' < j) \tag{5}$$

$$X_{ij} = 1 \text{ ならば } X_{i'j'} \neq 1 \, (i' < i, j < j') \tag{6}$$

式 (5) は，暗号文 c_i を平文 m_j に対応させた時，c_i より小さい暗号文は m_j より大きい暗号文に対応しない制約を表している．式 (6) は，暗号文 c_i を平文 m_j に対応させた時，c_i より大きい暗号文は m_j より小さい暗号文に対応しない制約を表している．LSAPに式 (5)，式 (6) の制約条件を加えた問題を順序保存割り当て問題と呼ぶこととする．

順序保存割り当て問題は $m \times n$ 行列 $C = (c_{ij})$, $(0 \leq c_{ij}, 1 \leq i \leq m, 1 \leq j \leq n)$ に対して，次のように定式化される．

$$\text{minimize} \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} X_{ij}$$

$$\text{subject to} \sum_{j=1}^{n} X_{ij} = 1, 1 \leq i \leq m$$

$$X_{ij} \in \{0, 1\}, 1 \leq i \leq m, 1 \leq j \leq n$$

$$X_{ij} = 1 \text{ ならば } X_{i'j'} \neq 1 \, (i < i', j' < j)$$

$$X_{ij} = 1 \text{ ならば } X_{i'j'} \neq 1 \, (i' < i, j < j')$$

3.2 順序保存割り当て問題を解くアルゴリズム

この問題を解くアルゴリズムを提案する．コスト行列の順序保存割り当て問題はコスト行列の部分行列の順序保存割り当て問題に帰着される．まず，提案手法のアルゴリズムを簡略化したアルゴリズムを説明する (Algorithm 2)．

Algorithm 2 Optimize

```
1: function Optimize(C):
2: Input:m × n コスト行列 C
3: Output:順序保存割り当て問題の解
4: if C が 1 行 then
5:     OUTPUT argmin C
6: else if C が n 次正方行列 then
7:     OUTPUT E_n
8: else
9:     A ← 0 _{m×n}
10:    A[1 : m − 1][1 : n − 1] ←Optimize(C[1 : m − 1][1 : n − 1])
11:    A[m][n] ←1
12:    B ← 0 _{m×n}
13:    B[1 : m − 1][1 : n] ←Optimize(C[1 : m − 1][1 : n])
14:    OUTPUT argmin_{X∈{A,B}} Σ_{i=1}^{n} Σ_{j=1}^{n} C_ij X_ij
15: end if
```

4から7行目の説明をする．コスト行列が1行の時は，その行の最小値が最小の割り当てとなる．コスト行列が正方行列の場合には，順序を保存する制約があるので，対角成分の割り当てが唯一の割り当てであり．対角成分の割り当てが最小の割り当てである．

次に，8行目以降の説明をする．コスト行列 C を $m \times n$ 行列として C の $i \times j$ 部分行列 $C_{[1:i][1:j]}$ の順序保存割り当て問題の解を $X_{[1:i][1:j]}$ とする．ただし，$M_{[c_1:c_2,r_1:r_2]}$ は行列 M の c_1 行目から c_2 行目，r_1 列目から r_2 列目の部分行列とする．このとき，C の順序保存割り当て問題の解の候補は次の A, B で与えられる．

$$A = \begin{pmatrix} X_{[1:m-1][1:n-1]} & 0 \\ 0 & 1 \end{pmatrix} \tag{7}$$

$$B = \begin{pmatrix} X_{[1:m-1][1:n]} 0 \end{pmatrix} \tag{8}$$

この性質より，C の順序保存割り当て問題を解くためには割り当て A, B に対応する目的関数の値を求めて比較し，小さい方を解として採用すればよい．

しかし，上記のアルゴリズムは重複する再帰呼び出しがあり，非効率的である．重複している再帰呼び出しを除いたアルゴリズムが実際の提案手法である (Algorithm3)．求めた部分行列の割り当て結果を記憶することで，再帰呼び出しの重複を回避している．実際の提案手法は，下記のようになる．また，$C[(i,j)]$ は C の i 行 j 列を表し，$C[(c_1, r_1), (c_2, r_2), \cdots, (c_n, r_n)]$ は $\sum_{i=1}^{n} C[(c_i, r_i)]$ を表す．

Algorithm 3 Optimize

```
1: function Optimize(C):
2: Input:m × n コスト行列 C
3: Output:順序保存割り当て問題の解
4: D(各部分行列の解を保存する． m × n の配列)
5: for i = 1 to n do
6:     if C[D[1][i-1]] ≤ C[(1,i)] then
7:         D[1][i] ← D[1][i − 1]
8:     else
9:         D[1][i] ← [(1, i)]
10:    end if
11: end for
12: for i = 2 to n do
13:    D[i][i] ← [(1, 1), · · · (i, i)]
14: end for
15: for i = 2 to m do
16:    for j = i + 1 to n do
17:        if C[D[i][j − 1]] ≤ C[D[i − 1][i − 1]] + C[(i, j)] then
18:            D[i][j] ← D[i][j − 1]
19:        else
20:            D[i][j] ← D[i − 1][j − 1] + [C[(i, j)]]
21:        end if
22:    end for
23: end for
24: OUTPUT D[m][n]
```

このアルゴリズムの計算量は $O(mn)$ である．

3.3 例

例として，下記のようなコスト行列が与えられた時の順序保存最適化問題を解く．

$$C = \begin{pmatrix} 12 & 11 & 13 & 17 & 19 & 5 \\ 8 & 9 & 4 & 7 & 12 & 5 \\ 7 & 8 & 6 & 5 & 4 & 2 \end{pmatrix} \tag{9}$$

まずは，$C_{[1:2,1:3]}$ の順序保存最適化問題を解く．

$$C_{[1:2,1:3]} = \begin{pmatrix} \boxed{12} & \boxed{11} & 13 \\ 8 & \boxed{9} & \boxed{4} \end{pmatrix} \tag{10}$$

$C_{[1:2,1:3]}$ の解を求める時に，$C_{[1:2,1:2]}$ の最小和と $C_{[1:1,1:2]}$ の最小和 $+ C_{[2,3]}$ を比較する．$C_{[1:2,1:2]}$ は正方行列なので順序保存割り当ては対角成分のみなので対角成分の割り当てが解である．$C_{[1:1,1:2]}$ は 1 行なので各成分の最小値の成分が解である．よって $C_{[1:2,1:3]}$ の解は (1, 1), (2, 2) と (1, 2), (2, 3) の 2 つの割り当てを比較し小さい方を解として採用する．ただし，(k, l) は k 行 l 列の対応とする．$\min(12 + 9, 11 + 4)$ となるから，最小の割り当ては，(1, 2), (2, 3) で最小値は 15 である．

$$C_{[1:2,1:4]} = \begin{pmatrix} 12 & \boxed{11} & 13 & 17 \\ 8 & 9 & \boxed{4} & \boxed{7} \end{pmatrix} \tag{11}$$

次に，$C_{[1:2,1:4]}$ の最小割り当てを求める．$C_{[1:2,1:3]}$ の最小和と $C_{[1:1,1:3]}$ の最小和 $+ C_{[2,4]}$ を比較する．$C_{[1:2,1:3]}$ の最小和は先ほど求めた割り当てである．$C_{[1:1,1:3]}$ の最小和は 1 行なので最小値の成分が解である．ゆえに，$C_{[1:2,1:4]}$ の解は (1, 2), (2, 3) と (1, 2), (2, 4) の 2 つの割り当てを比較すれば良い．よって $C_{[1:2,1:4]}$ の解は (1, 2), (2, 3) である．同様に $C_{[1:2,1:5]}$ の解を求めると (1, 2), (2, 3) となる．

$$C_{[1:3,1:4]} = \begin{pmatrix} \boxed{12} & \boxed{11} & 13 & 17 \\ 8 & \boxed{9} & \boxed{4} & 7 \\ 7 & 8 & \boxed{6} & \boxed{5} \end{pmatrix} \tag{12}$$

次に $C_{[1:3,1:4]}$ の解を求める．$C_{[1:3,1:3]}$ の最小和と $C_{[1:2,1:3]}$ の最小和 $+ C_{[3,4]}$ を比較する．$C_{[1:3,1:3]}$ は正方行列なので対角成分の割り当てが最小である．また，$C_{[1:2,1:3]}$ は先ほど求めた (1, 2), (2, 3) が最小の割り当てである．(1, 1), (2, 2), (3, 3) と (1, 2), (2, 3), (3, 4) を比較すると (1, 2), (2, 3), (3, 4) が解になる．$C_{[1:3,1:5]}$ は $C_{[1:3,1:4]}$ と $C_{[1:2,1:4]}$ が求まっているので計算ができる．$C_{[1:3,1:5]}$ と $C_{[1:2,1:5]}$ の解より C の解が求まり $(1, 2), (2, 3), (3, 6)$ が解である．

注意 1 順序情報を制約条件として組み込んでいるため，CDF は不要と思うかもしれない．スペースの都合により省略するが，数値実験では CDF を組み込んだ最適化関数の方が高い成功率を示している．詳細は，フルバージョンで報告する予定である．

4. 実験

[9] では，暗号文空間に対して，攻撃者が取得した暗号化データが十分多い時には，Cumulative 攻撃により生データを復元できることが示されている．今回の実験では，Cumulative 攻撃では復元が困難な状況に対する実験を行う．すなわち，暗号文の種類が多く暗号化データが少ない状況下に対して実験を行う．

4.1 用いるデータ

この実験では，UCI Machine Learning Repository Adult Data Set[14] の 48842 件を用いる．このデータセットは訓練データ 32561 件とテストデータ 16281 件から構成される．補助的なデータは訓練データからサンプルし，暗号化データはテストデータからサンプルしている．

年齢カラムを復元のターゲットとした．ここで，年齢カラムは，「17 歳」から「90 歳以上」である．図 1 に，年齢カラムのヒストグラムを記す．[9] では，データセットは異なるものの，年齢データの復元が行われているが，データが少ない時，年齢カラムの復元率が悪いという問題点があった．これより，2 つのアルゴリズムの比較する際に用いるデータとして年齢カラムが適切である．

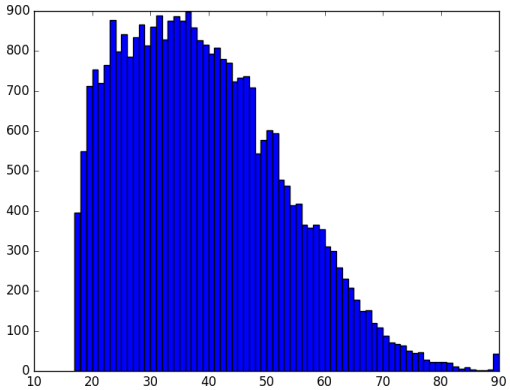

図 1 Adult Dataset 年齢カラム

以下の 2 つの攻撃シナリオに基づいた実験を行う．

(1) 補助的なデータが多く，暗号化データが少ないシナリオ (暗号化データ:1000 件，補助的なデータ:32561 件)

(2) 補助的なデータと暗号化データが少ないシナリオ (暗号化データ:1000 件，補助的なデータの数:1000 件)

現実的な脅威では補助的なデータが多く，暗号化データが少ない場合の方が一般的であると考えられる．なぜな

ら，攻撃者は公開されているデータを用いる方が容易だからである．しかし，[9] では過去の漏洩情報を用いるシナリオも考えれていて，その場合は補助的なデータが少なくなることが考えられる．今回は補助的なデータが少ない場合のシナリオについても実験を行った．

このデータベースの設定に基づいたデータベースを 500 個用意し，それらに攻撃を行い復元結果を評価する．シナリオ (1) での攻撃の評価は，暗号化データ 1000 件をテストデータからランダムにサンプリングして，データベースを 500 個生成する．そして，訓練データを補助的なデータとして用いることで 500 個のデータベースに対して攻撃を行う．シナリオ (2) での攻撃の評価は，暗号化データ 1000 件をテストデータからランダムにサンプリングして，補助的なデータを訓練データからランダムに 1000 件サンプリングして，データベースを 500 個生成する．最後に 500 個のデータベースそれぞれをある指標で評価して，全体の結果をグラフにした．

4.2 評価方法

今回は，アルゴリズムの 2 つの規準に評価を行った．1 つ目は [9] で行われた評価方法であり，暗号化データが正しく復元できた割合を評価する手法である．2 つ目は復元結果と生データをレーベシュタイン距離によって評価する方法である．

レーベシュタイン距離とは文字列の類似度を測る際に用いられる評価基準である．今回は復号結果と生データをそれぞれ文字列とみなしてレーベシュタイン距離によりこの 2 つの結果の類似度を測る．復元結果に数値の挿入，削除，置き換えの操作を行い生データと一致するまでの操作の回数を評価とする．レーベシュタイン距離の例を説明する．c_i を暗号文の中で i 番目に小さい暗号文，m_i を平文空間で i 番目に小さい平文とする．この時，$c_i \to m_i$ が正しい対応である．

復元結果として $c_1 \to m_2, c_2 \to m_3, c_3 \to m_4, c_6 \to m_5$ という対応が得られたとする．この時のレーベシュタイン距離は図 2 のように算出する．この場合，3 回の操作で正しい復元結果になるのでレーベシュタイン距離は 3 である．

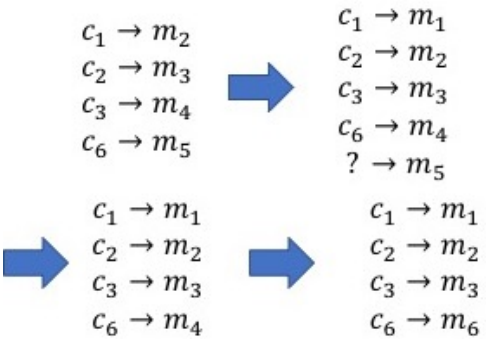

図 2 レーベシュタイン距離の例

この評価を新しく導入する理由は，次のような考えに基づく．[9] の評価方法では，データが完全一致の場合のみを考慮している．しかし，ある程度，正しい結果が得られれば攻撃が成功していると考える状況も多い．例えば，1 つずつ値がずれて復元された場合でも，多くの場合攻撃が有効であるとみなしてよい．そのような復元結果も高く評価する評価方法としてレーベシュタイン距離による比較も行う．

4.3 実験結果 (シナリオ 1)

図 3 に暗号化データが復元できた割合の比較を示す．提案手法は横軸の値が 80，縦軸の値が約 400 のところでプロットされている．これは提案手法によって 80%以上復元できたデータベースの数が 500 件中 400 件であることを示している．

提案手法は既存手法よりも 80%以下の復元率ではわずかに劣っている．しかし，85%以上の復元率では提案手法は性能が向上していることがわかる．既存手法では 85%以上復元できる件数が 0 件であるのに対して，提案手法では 400 件以上の復元ができており，大きく向上している．

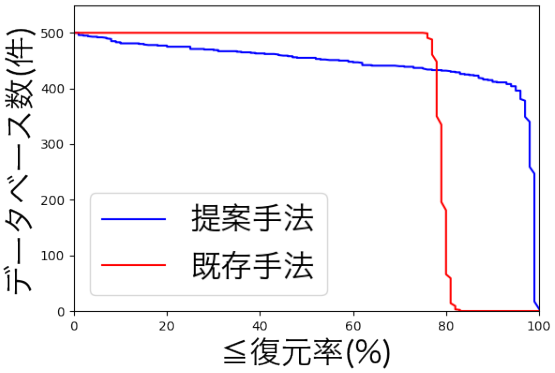

図 3 シナリオ 1 :(暗号化データ:1000 件, 補助的なデータ:32561 件)

80%以下の復元率では既存手法より提案手法の性能が悪化している原因を考察する．図 4 は，復元率を 10%ごとに区切り，データベースの個数を示したグラフである．目盛が x のビンは x 以上 $x+10$ 未満を示している．最後のビンのみ 100%復元できたデータベースの個数を示している．既存手法は 70%以上 90%未満復元できたデータベースの件数が提案手法に比べて多いことがわかる．これより，既存手法は順序を入れ替えた結果が得られることがあり，それによって順序が部分的に一致することがある．その分，提

案手法よりも良い結果が得られたと考えられる．

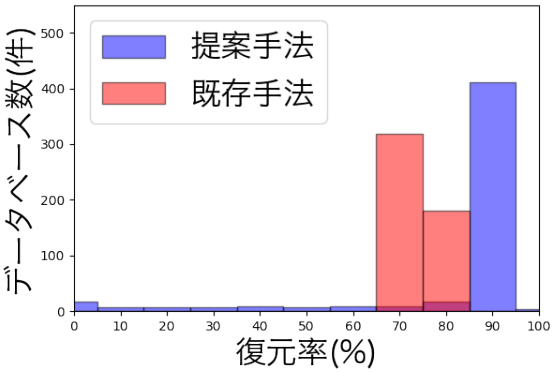

図 4　暗号化データ:1000 件，補助的なデータ:32561 件 (ヒストグラム)

データを低い復元率で復元されるよりも高い復元率で復元される可能性の方がはるかに脅威であるということを考え合わせると，提案手法は既存手法よりも性能が向上していると考える．

次にレーベシュタイン距離を用いた評価について説明する．レーベシュタイン距離の距離の平均は既存手法が 14.65，提案手法が 4.266 であった．図 5 に，既存手法と提案手法をレーベシュタイン距離を比較したグラフを記す．レーベシュタイン距離で比較した場合，提案手法が既存手法よりも復元結果と正しい結果の類似度が高いことがわかる．提案手法はレーベシュタイン距離で見た場合でも性能が向上している．これは，提案手法がある箇所から 1 つずつ値が異なるような復元結果になる場合があることが起因している．例えば，暗号化データの暗号文に最も小さい暗号文が含まれておらず，2 番目の暗号文を最も小さい平文に当てはめてしまった場合，そこから復元結果がずれてしまうような結果が得られ，その結果，復元結果が 0%になってしまう場合もある．しかし，1 つずつ値が異なっていてもデータによっては復元が成功していると捉えることもできる．そのような捉え方をした場合は提案手法は既存手法より優れていると考えられる．

4.4　（実験結果（シナリオ 2）

次に暗号化データと補助的なデータが少ない場合における，実験の結果について説明する．図 6 は復元率の結果を示している．既存手法よりも 80%以上の復元率では提案手法の性能が向上しいることがわかる．また，提案手法は補

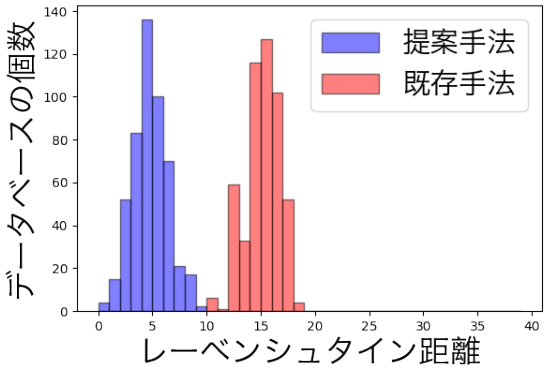

図 5　暗号化データ:1000 件，補助的なデータ:32561 件

助的なデータが少ない時の復元結果件数の減少が少ないこともわかる．図 7 は，復元率を 10%ごとに区切り，データベースの個数を示したグラフである．このグラフは提案手法は 70%以上 90%未満の復元できたデータベースの件数が既存手法に比べて劣っている．これは既存手法が部分的に順序を入れ替えた結果が得られ復元結果が部分的に一致する可能性が提案手法よりも多くなるからである．提案手法は既存手法に比べ 90%以上の復元率に関しては既存手法を上回る性能を示している．

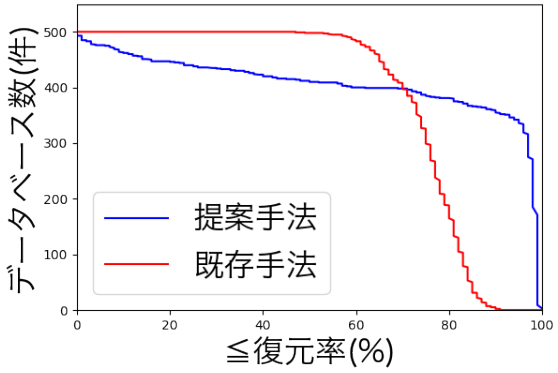

図 6　シナリオ 2（暗号化データ:1000 件，補助的なデータ:1000 件）

次にレーベシュタイン距離による比較結果を確認する．レーベシュタイン距離の平均は既存手法が 14.43 提案手法

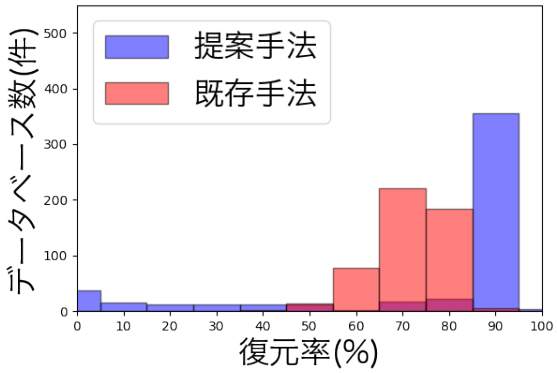

図 7　暗号化データ:1000 件，補助的なデータ:1000 件 (ヒストグラム)

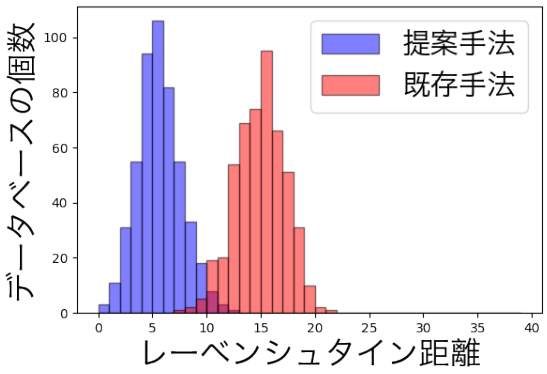

図 8　暗号化データ:1000 件，補助的なデータ:1000 件

が 5.14 である．図 8 はレーベシュタイン距離による比較結果を示している．レーベシュタイン距離で見た場合，提案手法は既存手法に比べて性能が大きく向上していることがわかる．

5. 結論

我々は，順序保存割り当て問題を導入し，この問題を解くアルゴリズムを提案し，Cumulative 攻撃の改良を行なった．順序制約をつけることにより，85%以上の復元率においては既存手法を上回ることを実験により示した．さらに，レーベシュタイン距離による評価では既存手法を上回る性能を示した．提案アルゴリズムは，補助的なデータが少ない時にも有効なアルゴリズムとなっている．

謝辞　本研究の一部は，JST CREST JPMJCR1302 の支援および JSPS 科研費 26540003 の助成を受けて行われた．

参考文献

[1] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu. Order preserving encryption for numeric data. In Proc. of SIGMOD, pp. 563–574, 2004.
[2] A. Boldyreva, N. Chenette, Y. Lee, and A. O'neill. Order-preserving symmetric encryption. In Proc. of EUROCRYPT, pp. 224–241, 2009.
[3] A. Boldyreva, N. Chenette, and A. O' Neill. Order-preserving encryption revisited: Improved security analysis and alternative solutions. In Proc. of CRYPTO, pp. 578–595, 2011.
[4] P. Karras, S. Malhotra, R. Bhatt, A. Nikitin, D. Antyukhov, and S. Idreos. Adaptive indexing over encrypted numeric data. In Proc. of SIGMOD, pp. 171–183, 2016.
[5] D. Boneh, K. Lewi, M. Raykova, A. Sahai, M. Zhandry, and J. Zimmerman. Semantically secure order-revealing encryption: Multi-input functional encryption without obfuscation. In Proc. of EUROCRYPT, pp. 563–594, 2015.
[6] N.Chenette, K. Lewi, S. A. Weis, D. J. Wu. Practical Order-Revealing Encryption with Limited Leakage. In Proc. of FSE, pp. 474–493, 2016.
[7] K.Lewi, D. J. Wu. Order-Revealing Encryption: New Constructions, Applications, and Lower Bounds. In Proc. of ACM CCS 2016, pp. 1167–1178, 2016.
[8] R. A. Popa, C. Redeld, N. Zeldovich, and H. Balakrishnan. CryptDB: Protecting confidentiality with encrypted query processing. In Proc. of SOSP 2011, pp. 85–100, 2011.
[9] M. Naveed, S. Kamara, and C. V. Wright. Inference attacks on property-preserving encrypted databases. In Proc. of ACM CCS 2015, pp. 644–655, 2015.
[10] C. Horst, R. Kikuchi, K. Xagawa Cryptanalysis of Comparable Encryption in SIGMOD'16. In Proc. of SIGMOD'17, pp. 1069–1084, 2017.
[11] F. Betl Durak, T. M. DuBuisson, and D. Cash. What else is revealed by order-revealing encryption? In Proc. of ACM CCS, pp. 1155–1166, 2016.
[12] H. W. Kuhn. The Hungarian method for the assignment problem. Naval Research Logistics Quarterly, 2, 1955.
[13] J. Munkres. Algorithms for the assignment and transportation problems. Journal of the Society for Industrial and Applied Mathematics, 5(1), 1957.
[14] UCI Machine Learning Repository: Adult Data Set https://archive.ics.uci.edu/ml/datasets/adult