4V-04

索引を用いた秘匿検索における安全性の高い複数演算の連携法

秋山 賢人 † 渡辺 知恵美 ‡ 北川 博之 ‡

† 筑波大学情報学群情報科学類 ‡ 筑波大学システム情報系

1 はじめに

近年，Database as a Service（DBaaS）を利用したデータの検索サービスが普及しつつある．DBaaS を用いる大きな利点は，サーバの常時稼働及びデータベース管理，高速なアクセスの保障等のコスト及び責任のかかるタスクを委託できる点にある．

その一方で，DBaaS のサービスプロバイダ及び利用者は，保存するデータや利用者自身が発行するクエリに対して個人情報が漏洩するかもしれないというリスクがある．このようなリスクに対応するため，近年は暗号化データベースシステムに関する研究が盛んに行われている．暗号化データベースシステムではデータを暗号化して DBaaS で管理し，検索可能暗号スキームを利用してデータを暗号化したまま問合せ処理を行う．

篠塚らは [1] において，機密性を守りつつ高速な検索処理を可能にする OSIT-bs（Oblivious Secure Index Traversal-binary search）フレームワークを提案した．OSIT-bs では索引の一部を暗号化したままクライアントとサーバに分割して配置し，サーバ及びクライアントが完全な索引の情報を知らないまま索引の探索を行うことができる．また，データや索引を秘匿したまま問合せを行えるほか，クエリの内容自体も秘匿可能である．この研究では，単一の索引を探索することを想定しているため，1 回の問合せでは「SELECT * FROM R WHERE age > 40」などの一つの属性を検索条件とした範囲検索または完全一致検索にのみ対応していた．

しかし，「SELECT * FROM R WHERE age > 40 AND salary < 200,000」などの連言節を含んだ複数の条件の場合は 1 回の問合せで処理することができず，各々の条件式をクライアントで評価し，結果の論理積をクライアントで評価する必要がある．これはクライアントの負荷が大きく，問合せの中間結果がクライアントに漏洩するという安全性の問題も存在する．

そこで，本研究では先行研究をもとに，連言節で接続された複数の条件を含む問合せに対し，問合せの中間結果をクライアントに明かさずに求める手法を提案する．

A Secure Search Method for Multiple Operations with Data Interoperability using Index

Kento Akiyama†, Chiemi Watanabe‡, Hiroyuki Kitagawa‡

†College of Information Science, University of Tsukuba

‡Faculty of Engineering, Information and Systems, University of Tsukuba

本研究を実現するために，Wong らの手法 [2] で導入されている鍵更新演算を利用し，属性ごとに異なる鍵で暗号化されたレコード ID を比較してクライアントに中間結果を与えることなく複数条件による検索を行う．

2 先行研究

篠塚らはデータとクエリ双方のプライバシを保護した高速な秘匿検索フレームワークである OSIT-bs[1] を提案している．このフレームワークは，データ所有者 (Data Owner: DO) がデータを暗号化して DBaaS のサービスプロバイダ (Service Provider: SP) に管理を委託し，クライアントは DO から検索の権限をもらって問合せを行うという状況を想定しており，満たすべきプライバシ要件は以下の通りである．

(1) サービスプロバイダは暗号化されたデータの内容及びクエリ履歴から元の値を推測できない．

(2) クライアントは発行した問合せに対する結果以外の情報を得ることができない．

この手法の特徴は DO がデータとともに検索用のデータベース索引を作成し、索引を暗号化してサーバとクライアントに分割して配置するところにある．サーバは暗号化された索引のノードを所有している．また，クライアントは，索引のノード間関係に関する情報を DO から与えられる．各ノードに関してはサーバ上のアドレスのみ知らされており，ノード自体を取得することはできない．このようにサーバ・クライアント双方が完全な索引の情報を知らずに索引を探索することができる．

以下，索引の探索方法について述べる．OSIT-bs において SP は索引とデータを保管するデータサーバと大小比較を行う計算サーバの 2 台のサーバを提供する．索引の探索にはノードの値とクエリ値の大小比較を繰り返し行う必要がある．ここではサーバ上のあるノードの値 k とクライアントで発行されたクエリ値 q の比較を行う手順について述べる．なおサーバ上のノードには Paillier 暗号 $E(\cdot)$ で暗号化された値 $E(k)$ が保存されており，そのノードのサーバ上のアドレスを $h(k)$ とする．まず，クライアントは q にランダム値 r を加えた q ' を求める．次に $E(r)$ を求めてデータサーバに送り，$h(k)$ にあるノードの値 $E(k)$ に $E(r)$ を乗算して計算サーバに送るよう要求する．Paillier 暗号は加法準同型性をもつため $E(k) \times E(r) = E(k+r)$ となる．次にクライアントは q ' を計算サーバに送り，$E(k+r)$ と q' の大小比較結果を要求する．大小比較には $E(k+r)$ を暗号化したまま比較結果を求めるため，GT-SCOT プロトコル [3] を用いている．GT-SCOT は大小比較を行う

紛失通信プロトコルであり，比較結果も暗号化されてクライアントに送られるためサーバは比較結果を知ることができない．クライアントは比較結果を復号化して大小比較結果を求める．これを繰り返すことで索引の探索が実現できる．なお，データサーバで保存されているリーフノードにはノードに該当するレコードの暗号化されたアドレスリストが保存されている．クライアントは索引をリーフノードまで辿り，該当レコードの ID リストを入手し，データサーバからレコードを取得する．

OSIT-bs は 1 回の問合せにつき 1 つの索引の探索を想定している．この手法の問題は複数の索引を探索する必要がある問合せを安全に評価することができないことである．例えば「WHERE age $>$ 40 AND salary $<$ 200,000」のような条件の問合せを評価する場合，OSIT-bs では「age」と「salary」各索引に対して探索を行い，探索で得たアドレスリストを復号してサーバからレコードを入手する．クライアントは各結果のレコードから両方の条件を満たすレコードを最終結果として得る．この場合クライアントは各々の条件を満たす結果を知ることができ，プライバシ要件 (2) を満たさない．

3 提案手法

提案手法では連言節で接続された複数の条件を含む問合せに対し，クライアントに中間結果を明かさずに問合せを行う手法を提案する．例として二つの条件 A, B に対して A and B の結果を求める場合を考える．索引の探索が終了した時点で，条件 A,B それぞれに該当するレコード ID のリストはそれぞれの鍵 k_a, k_b で暗号化されている．それらを $R_A = \{E_{k_a}(a_1), ..., E_{k_a}(a_n)\}, R_B = \{E_{k_b}(b_1), ..., E_{k_b}(b_m)\}$ とする．暗号プロトコルは Wang ら [2] のスキームを用いている．このスキームは加法準同型性を持つほか，暗号化したまま鍵を更新する関数が定義されている．なお鍵 k_a から k_c への鍵更新関数をここでは $ChangeKey(E_{k_a}(a), k_a, k_c)$ とする．

提案手法のアルゴリズムを Algorithm 1 に示す．出力結果 $R_C = \{E_{k_c}(s_1), ..., E_{k_c}(s_n)\}$ は R_A の要素数と同数であり，レコード ID a_i が R_B に含まれる場合 $s_i = a_i$ となり，含まれない場合はレコード ID のドメイン外の値となる．R_C はクライアントで復号化し，レコード ID のドメイン内の値が A and B を満たすレコードとなる．Algorithm 1 の 3 行目の S_i を求める式では，まず A_iと B_j の鍵を共通の鍵 k_cに変更した後 A_iと B_j の差を求めている．ID が一致すれば 0 となり，そうでなければ 0 以外の値となる．これを B_0から B_mの値に対して総積を求めると，A_iのレコード ID が R_Bに含まれる場合のみ値は 0 となり，$S_i = A_i$ となる．これらの計算は暗号化したまま全て行うことができる．

4 評価実験

本稿では，提案手法の性能評価のために，Wong らの手法 [2] と比較を行う．提案手法はサーバに比較結果を明かさず，索引を使用している．[2] はサーバで複数演算を安全に行えるが，サーバに比較結果を明かしてしまうという問題があることから比較対象とした．実験では「WHERE age $>$ 40 AND salary $<$ 200,000」という条件の問合せに対し，各データ数における平均クエリ応答時間の比較を行った．結果を図 1 に示す．提案手法ではレコード数が増えると [2] よりも遅くなることがわかるが，これは条件を満たすレコード ID の増加に伴う比較回数の増加によるものだと考えられる．

Algorithm 1 Logical Product (R_A, R_B)

Input: $R_A = \{A_1, ..., A_n\} = \{E_{k_a}(a_1), ..., E_{k_a}(a_n)\}$,
$R_B = \{B_1, ..., B_m\} = \{E_{k_b}(b_1), ..., E_{k_b}(b_m)\}$

Output: $R_C = \{S_1, ..., S_n\} = \{E_{k_c}(s_1), ..., E_{k_c}(s_n)\}$

1: **for** each $A_i \in R_A$ **do**
2: r_i はレコード ID のドメインを超えるランダム値
3: $S_i = E(r_i) \times \prod_{j=0}^{m}(ChangeKey(A_i, k_c) - ChangeKey(B_j, k_c)) + A_i$
4: **end for**

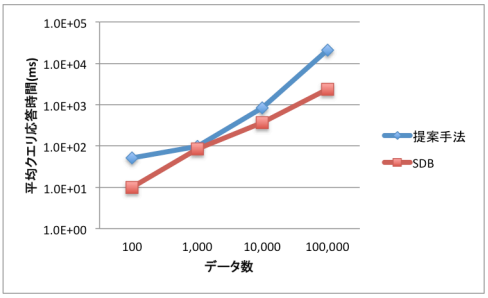

図 1 各データ数における平均クエリ応答時間

5 まとめ

本稿では，複数属性を含む問合せに対し，中間結果をクライアントに知られずに求める手法を提案した．また，実験結果からレコード数が増えると比較回数が膨大になるため検索処理に時間がかかることが示された．

今後の課題としては，計算を高速に行えるように比較演算の改善を行いたい．

参考文献

[1] 篠塚 千愛, 渡辺 知恵美, 北川 博之. DaaS 環境におけるデータとクエリ双方のプライバシ保護を実現する効率的な秘匿検索. DEIM Forum 2015 G2-6 (2014).
[2] W. K. Wong et al. Secure Query Processing with Data Interoperability in a Cloud Database Environment. Proceedings of the 2014 ACM SIGMOD international conference on Management of data, pp. 1395-1406 (2014)
[3] I. E. Blake et al. Strong Conditional Oblivious Transfer and Computing on Intervals. Advances in Cryptology ASIACRYPT 2004, pp. 515-529 (2004).