ブルームフィルタを用いたプライバシ保護検索法

理学専攻　情報科学コース　新井裕子（指導教員：渡辺 知恵美）

1　はじめに

Database as a Service(DaaS) が注目される中，プライバシの関心が高まっている．DaaS におけるサーバ管理者はデータに対して第三者であるため，不十分な管理を行ったり，データを悪用する可能性がある．サービス利用者の個人情報を保護するため，データを暗号化し，プライバシを保護しつつ，暗号データに付加された索引を使って問合せを実現する研究が行われてきた．本稿では，ブルームフィルタを用いたプライバシ保護検索法とログを利用した攻撃に対する防御法を提案する．本手法の特徴は (1) 属性毎ではなくタプル毎に索引を構成し，元テーブルのスキーマを隠蔽すること (2) 数値を文字列の集合に置き換え，文字列属性と同様に大小比較を行うことで，検索条件や対象属性を秘匿すること (3) 偽の問合せを発行してログを攪乱し，統計情報を利用した攻撃を困難にすることである．

2　DaaS におけるプライバシ保護検索

図 1 に DaaS における一般的なプライバシ保護検索の流れを示す．DaaS ではデータベース管理者が攻撃者となりうる．DaaS における攻撃モデルを以下に示す．

- Direct Attack：DB 上のデータを直接盗みとる
- Indirect Attack：ログから統計情報を盗みとる
- Memory Attack：メモリ上のデータを盗みとる

これらの攻撃からデータを保護するために，信頼できるサーバまたはクライアント側でデータや問合せを暗号化したのちサーバに送信する (図 1)．サーバ側では，暗号化したデータに対して問合せを行うため，通常の問合せのように一回で正確な結果を返すことは難しい．そこでサーバ側では仮の結果を求め (図 1:1$^{st}query$)，信頼できるサーバで再度問合せを行い (図 1:2$^{nd}query$)，不正解データを含まない正確な解を得る．サーバ側には 1$^{st}query$ の情報が蓄積される (図 1:Query log)．

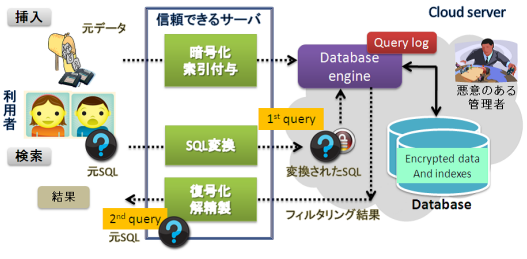

図 1: DaaS におけるプライバシ保護検索の流れ

これまで提案されてきたプライバシ保護検索手法は，(1) 数値属性に対する比較演算手法 [1] と (2) 文字列属性や文書に対するキーワード検索手法 [2] に分類できる．リレーショナルデータベースに対してプライバシ保護検索を実現する場合，上記のいずれかの手法を用いて属性毎に異なる方法でデータを変換する必要があった．そのためテーブルのスキーマ構成や検索対象となる属性や検索の種類を攻撃者から隠すことができない．

3　ブルームフィルタによるプライバシ保護検索

3.1　基本手法

我々はブルームフィルタを用いて文字列属性と数値属性に対して同様に索引を構成し問合せを行う [4]．図 2 にタプルからブルームフィルタ (以下 bfindex) を生成する流れを示す．まずタプル t から検索語の集合 W(t)=w_0,...,w_n を生成する．各語 w_i は属性名と属性値からなる．属性値が文字列である場合，単語毎や n-gram を使って文字列属性値を分割し，":"で属性名と繋げて，検索語とする．検索語全てに対して HMAC(鍵付きハッシュによるメッセージ認証関数) を複数適用し，ブルームフィルタにビットを立てる．検索の際も同様の手順で検索条件から検索語を生成し，各 bfindex に対してブルームフィルタテストを行う．

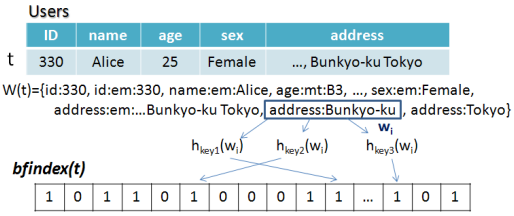

図 2: 索引構成の流れ

3.2　数値からの語の生成

我々は文献 [3] で数値から検索語を生成する方法を提案した．数値属性のドメインを複数のバケットに分割し，該当するバケット番号を属性名と合わせて検索語とする．図 3 に 25 と 38 に対する検索語の集合を示す．ドメイン [0,100] を 10 個に分割しバケット番号 B0,B1,...,Bb を割り当てる．バケットの上限が値 v より小さい場合は [属性名:lt:バケット番号]，バケットの下限が値 v より大きい場合は [属性名:mt:バケット番号]，それ以外は [属性名:em:バケット番号] という検索語となる．条件が"v < 35"の場合，35 が含まれるバケット (B4) の左隣のバケット (B3) に注目し，[属性名:lt:B3] を検索語としてブルームフィルタテストを行う．条件が"v < 35"の場合，35 が含まれるバケット (B4) の右隣のバケット (B5) に注目し，[属性名:mt:B5] を検索語とする．さらに，文献 [3] ではドメインの分割を多段階に行うことで，検索語の個数を抑え，データの挿入時間を減らす改良法を提案し実験を行った．挿入する数値のドメイン [0,1000]，分割数 10，索引の段階数 3，タプル数 1000 としたとき，改良後のデータ挿入時間は改良前に比べて 1/40 となり大幅に減少した．

-∞ 0 10 20 30 40 50 60 70 80 90 100 +∞

	B0	B1	B2	B3	B4	B5	B6	B7	B8	B9	Ba	Bb
25	lt:B0	lt:B1	lt:B2	em:B3	mt:B4	mt:B5	mt:B6	mt:B7	mt:B8	mt:B9	mt:Ba	mt:Bb
38	lt:B0	lt:B1	lt:B2	lt:B3	em:B4	mt:B5	mt:B6	mt:B7	mt:B8	mt:B9	mt:Ba	mt:Bb

図 3: 数値属性の変換例

3.3 2 段階ハッシュによるより安全な検索

基本手法では，同じ検索語をもつタプルから同じ位置にビットが立つ bfindex が生成されるため，ビットパタン分析により何らかの情報が漏れる可能性があった．そこで文献 [3] では，基本手法で一度ハッシュ関数を求めた後，タップルを丸ごと暗号化した etuple を鍵として再度 HMAC を適用する．これにより，同じ検索語をもつタプルでも異なる場所にビットが立った bfindex が生成されるため，攻撃者はビットパタンから元データの特徴を推測することはできない．しかしこの手法では，各タプルに対して 2 回 HMAC を適用するため検索時間がかかるという問題がある。文献 [4] における実験では Google App Engine でタプル数 1000，検索用のハッシュ値数 5，検索に対する正解率 60% としたとき，0.4 秒かかることがわかった．タプル数 20000 以上になるとタイムアウト (8 秒) せずに回答できる保証がない．そこで文献 [5] では検索時間に配慮して基本手法を採用し，ビット列にノイズを加えることでビットパタンを攪乱する方法を提案している．

3.4 クエリログを利用した攻撃とその攪乱

データベース管理者は攻撃の手段として，Cloud サーバに残るログの統計情報を利用できる．また管理者は (1) 業務知識：仕事環境により得られる知識と (2) 一般的な知識：一般常識やニュース，を持つ．Cloud サーバに発行された問合せに含まれるハッシュ値の分布が極端な場合，これらの知識と組み合わせることで，ある検索語に対するハッシュ値の集合を突き止めることが可能であるため，文献 [6] にて偽の問合せを発行しログを攪乱する方法を提案した．基本的には，各ハッシュ値の出現数の差が一定数以上に達したときのみ偽の問合せと発行する．図 4a に偽問合せの生成の流れを示す．入力値 $v_0,..,v_n$ はサーバに発行された問合せに含まれるハッシュ値の集合である．RealDiff(図 4a の*1) は出現数の差の最大値であり，MaxDiff(図 4a の*2) は各ハッシュ値の出現数の差の上限であり利用者が定めるものとする．なお一様または正規分布に従う乱数を偽のハッシュ値として使う．

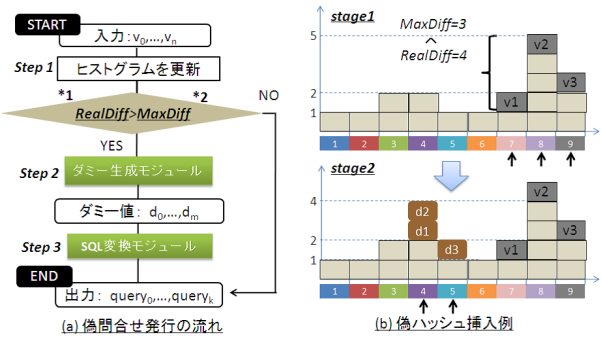

図 4: 偽問合せ生成の手順と偽ハッシュ値の挿入例

- step1:サーバに発行する問合せからハッシュ値を抽出して入力値とし，ヒストグラムを更新する
- step2:RealDiff が MaxDiff より大きい場合，一様または正規分布に従う偽のハッシュ値を生成する
- step3:偽のハッシュ値から複数の問合せを作成する

図 4b に正規乱数に従う偽ハッシュ値の挿入例を示す．v1,v2,v3 はユーザが発行した元問合せの検索語から生成された真ハッシュ値である．(図 4b では'7,8,9'(stage1 の矢印))．stage1 で RealDiff ＞ MaxDiff となるため，'4,4,5' をダミーとして挿入する (stage2 の矢印)．

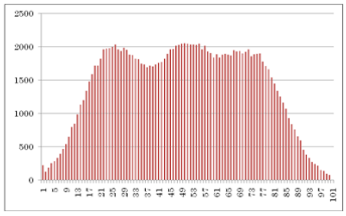

図 5: 偽ハッシュ値挿入後の分布

平均 50 分散 10 の正規分布に従う値を 5 個ずつ 1 万回入力し，2 種類の正規分布 (平均:v/2 と (3*v)/2，分散:10) に従うダミー値を挿入した結果の分布を図 5 に示す．v は出現数の最大値をとる値を示す．縦軸は出現数，横軸はハッシュ値である．攪乱後の分布から有用なハッシュ値の集合を特定するのは困難である．ダミーの個数は 90239 個となった．

4 まとめと今後の課題

本稿では，ブルームフィルタを用いて文字列属性と数値属性に対して同様に問合せを行えるプライバシ検索法を示した．また，数値属性の索引構成，2 段階ハッシュによる安全な検索，クエリログを利用した攻撃とその防御への取り組みを示した．DaaS 環境上に検索システムを実装し，検証することが今後の課題である．

参考文献

[1] Hacigumus H, et al:Executing sql over encrypted data in the database-service-provider model, Proceedings of the 2002 SIGMOD International Conference, pp.216-227.

[2] Boneh D, et al: Public Key Encryption with Keyword Search, Proceedings of EUROCRYPT, vol.3027 LNCS, pp.506-522 (2004)

[3] 新井裕子，渡辺知恵美: データベースアウトソーシングにおけるプライバシ保護に考慮した範囲検索法, 日本データベース学会論文誌，Vol.7, No.1, pp7-12，(2008).

[4] Watanabe C. and Arai Y.: Privacy-Preserving Queries for a DAS model using Two-Phase Encrypted Bloomfilter, Proceedings of International Conference on Database Systems for Advanced Applications, vol.5463 LNCS, pp.491-495 (2009)

[5] 渡辺知恵美，新井裕子，天笠俊之：ブルームフィルタを用いたプライバシ保護検索における攻撃モデルとデータ撹乱法の一検討，日本データベース学会論文誌 Vol.8, No.1, pp000-000 (2009)

[6] Arai Y. and Watanabe C.: Query Log Perturbation Method for Privacy Preserving Query, ICUIMC (2010).