D-002

XMLデータの暗号化に対応した安全な検索方法の提案

On a Secure Search Method for Encryped XML Data

中村 伸一†　　山本 博章‡
Shin'ichi Nakamura　　Hiroaki Yamamoto

1．はじめに

近年，ネットワークの信頼性の向上や回線の高速化に伴い，Amazon SimpleDB [1]等で Database as a Service（以下、DAS と略す）の運用が始められている．DAS はネットワークを経由してデータベースをアウトソースするサービスのことであり，データベース専用のシステムを用意することでデータベースの高信頼化や高性能化，多くのデータベースを一括管理することでサービスの低コスト化を期待することができる．

しかし，DAS の運用についてセキュリティ上の課題がいくつか存在し，その一つにデータベースのファイルが，アウトソース先のデータベース管理者に持ち出されることによる情報漏えいがある．

この課題を解決するためには，ファイルを暗号化する方法が有効であるが，さらに暗号化したままで検索することができれば，データベース管理者やメモリを盗聴することによるファイル上のデータの持ち出しにも対応することができる．

暗号化されたデータの検索に関する研究について，例えば，[2]は暗号化された文字列の検索手法を，[3]は暗号化された XML データの検索手法について述べている．

今回は，XML データの要素を暗号化されたままで検索する手法を提案する．手法は，検索元 XML データの各要素を暗号化したデータベースを作成しサーバに保存する．次に，クライアントで検索する XML データの根の要素名を暗号化し，サーバでデータベース上の暗号化された要素名と一致する要素を検索する．次に，クライアントは一致した要素を根とする木を構成する要素をサーバから受け取る．最後に，クライアントは要素を復号し，検索する XML データで検索を行なう．

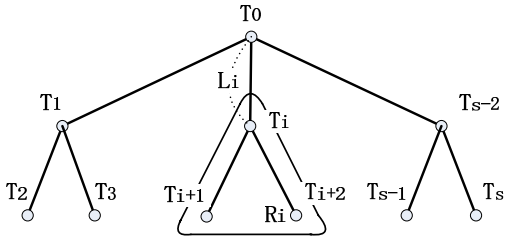
サーバから要素を根とする木を構成する要素を受け取る場合，サーバ側で根以外の要素も検索することで，サーバからの通信量を少なくする改善も行なっている．

本文の構成は，2 章で暗号化処理，3 章で検索アルゴリズム，4 章で実験，5 章で評価について述べる．

2．XML の暗号化処理

2.1　XML の保存形式

XML データを検索するための保存形式として，RDB に XML の構造をマップする形式，XML の構造をそのままデータとする形式がある．仕組みが単純であることから，XML の構造をそのままデータとする形式の一つである[4]で提案された保存形式を採用している．この保存形式は，データが XML データの深さ優先探索順に直線的に並べられ，データに要素に関連する情報が保存されるようになっている．



$\langle T_0, L_0, R_0\rangle, \ldots, \langle T_i, L_i, R_i\rangle, \ldots, \langle T_s, L_s, R_s\rangle$

T_i：i番目の要素名
L_i：i番目の要素のXML文書上の木の深さ
R_i：i番目の要素を根とした木を構成する最後の要素

図 1 データ形式

2.2　XML の暗号化処理

2.1 で説明した保存形式を暗号化する場合，同じデータを暗号化すると同じ暗号列となり，データの出現傾向を推測される可能性がある．そのため，暗号化を行なう際には，サーバとクライアントが暗号化データをやり取りするため共通して持つ鍵 K を用意し，K とデータの位置を元にハッシュ関数 H でハッシュ化を行ったものを，データを暗号化する関数 E の鍵とすることにより，同じ暗号列が発生しないようにしている．

平文　：$\langle T_0,L_0,R_0\rangle,\ldots,\langle T_i,L_i,R_i\rangle,\ldots,\langle T_s,L_s,R_s\rangle$
暗号文：$\langle E_{T0},E_{L0},E_{R0}\rangle,\ldots,\langle E_{Ti},E_{Li},E_{Ri}\rangle,\ldots,\langle E_{Ts},E_{Ls},E_{Rs}\rangle$

E_{Ti}：E(H(K,i), T_i)
E_{Li}：E(H(K,i), L_i)
E_{Ri}：E(H(K,i), R_i)

3．暗号化 XML の検索

今回提案する手法は，２つの XML データ T,P に対し，T に出現する P をすべて検索する問題を扱う．なお，T をターゲット XML データ、P をクエリ XML データと呼ぶ．

† 信州大学大学院工学系研究科
‡ 信州大学工学部

3.1 検索アルゴリズム

暗号化された XML データの検索について，以下の手順で行なう．

（1）T を 2.1 で説明した保存形式に変換し，2.2 で説明した手順で各要素の情報を暗号化する．
（2）P の根の要素名を，2.2 で説明した手順で T の要素数分暗号化する．
（3）（1）と（2）を比較し，要素名が一致する T のデータを抽出する．
（4）（3）を復号し，（3）を根とした木を構成する要素の範囲を得る．P の葉を，2.2 で説明した手順で要素の範囲分暗号化する．
（5）（4）の要素の範囲の中に，暗号化された P の葉が含まれる T のデータを抽出する．
（6）（5）を復号し，P が含まれるか検索する．

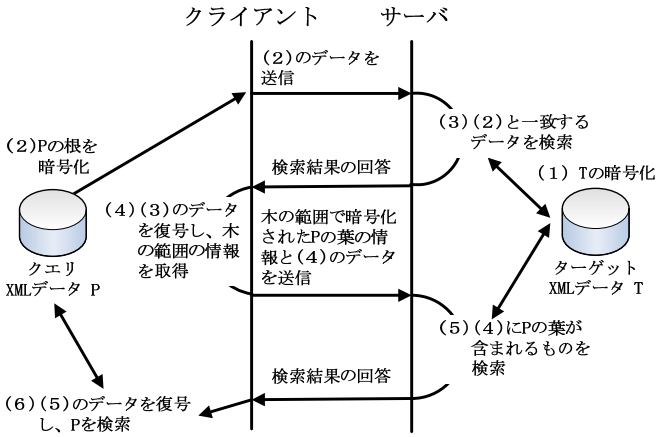


図 2 検索アルゴリズム

3.2 XML データ検索アルゴリズム

3.1 の（6）で P の検索を行なっているが，この検索アルゴリズムについて，仕組みが単純であることから，要素名を逐次比較する方法を採用している．

4. 実装実験

本手法の実行について，どの程度時間がかかるか実際にプログラムを作成し，実験を行なった．実験で用いた環境は，CPU は Intel Xeon 3.2GHz，メモリ 4GByte，プログラム言語は Perl とした．

なお、検索元のデータは XML Benchmark Project[5]で開発された xmlgen を利用し，25Kbyte，50Kbyte，100Kbyte，200Kbyte のデータで実験を行なった．検索する XML データは，XML Benchmark Project 内の XQuery の問 1 を修正したもの（/people/person/name の検索）を使用し，ハッシュアルゴリズムは MD5，暗号アルゴリズムは Blowfish を使用した．

実行した結果は以下の通りとなった．作成時間と検索時間は，データ量に応じて比例しており，データの検索時間は，データの作成時間の約 30～40%程度となっている．

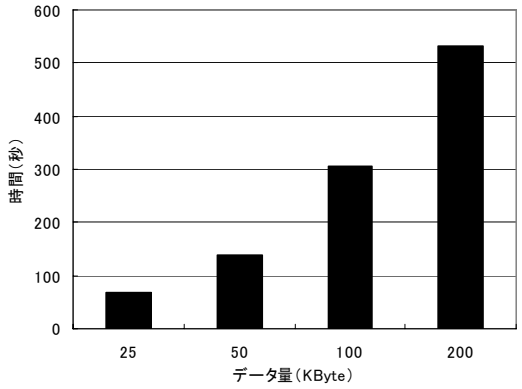


図 3 暗号化データの作成時間

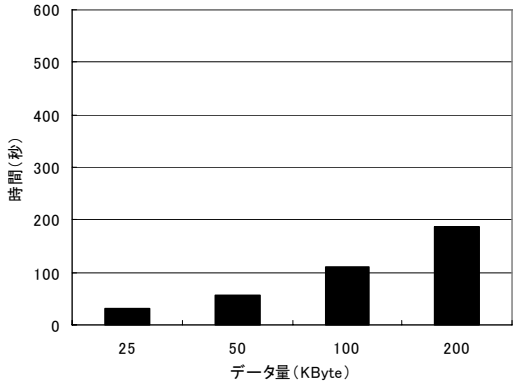


図 4 暗号化データの検索時間

5. まとめと今後の課題

今回は，要素が暗号化された XML データの検索方法について提案を行い，提案した手法で実装したプログラムの処理時間の実験を行った．今後の課題は，検索時間の高速化や，XML 構造の推測防止があげられる．検索時間の高速化について，3.1 の（2）に処理の多くの時間が使われている．そのため，暗号化する要素を減らすためにインデックスを利用することで，高速化する方法が考えられる．また，XML 構造の推測防止について，現在 Ri は正しい位置を保存しているため，XML 構造が推測される可能性がある．そのため，Ri を木の範囲を含む大きな値とすることで，推測されることを防ぐ方法が考えられる．

参考文献

[1] http://www.amazon.com/gp/browse.html?node=342335011
[2] 山﨑康博, 山本博章,“暗号化データに対する安全な全文検索技術について”, 暗号と情報セキュリティシンポジウム, 2006.
[3] Hui (Wendy) Wang, Laks V.S. Lakshmanan, “Efficient Secure Query Evaluation over Encrypted XML”, VLDB ‘06, 2006.
[4] P. Zezula, G. Amato, F. Debole, and F. Rabitti, “Tree Signatures for XML Querying and Navigation”, In Proceedings of the XML Database Symposium, 2003.
[5] http://monetdb.cwi.nl/xml/