2006－DBS－140（Ⅰ）（26）
2006／7／12

XMLデータベースへの型推論を用いた攻撃に対する安全性検証

高須賀 史和† 　橋本　健二† 　石原　靖哲† 　藤原　融†

† 大阪大学大学院情報科学研究科
〒 565–0871 吹田市山田丘 1–5
E-mail: †{fmkz-tak,k-hasimt,ishihara,fujiwara}@ist.osaka-u.ac.jp

あらまし　推論攻撃とは，ユーザが許可されている問合せとその実行結果から，許可されていない問合せの実行結果 (機密情報) を推論しようとすることである．本稿では，ユーザが許可されている複数の問合せとその結果から，型推論を用いて機密情報の候補を絞り込む攻撃を考える．そして XML データベースへの型推論を用いた攻撃に対する安全性を形式的に検証する手法を提案する．

キーワード　データベースセキュリティ，XML，推論攻撃，型推論

Security Verification against Attacks Using Type Inference on XML Databases

Fumikazu TAKASUKA†, Kenji HASHIMOTO†, Yasunori ISHIHARA†, and Toru FUJIWARA†

† Graduate School of Information Science and Technology, Osaka University
1–5 Yamadaoka, Suita, Osaka, 565–0871, Japan
E-mail: †{fmkz-tak,k-hasimt,ishihara,fujiwara}@ist.osaka-u.ac.jp

Abstract　Inference attacks mean that an attacker tries to infer the execution result of a query unauthorized to the attacker (i.e., secret information) from the execution results of queries authorized to the attacker. In this manuscript, attacks using type inference are considered, where an attacker narrows the candidates for the secret information using authorized queries and their execution results. Then, a method of verifying the security against attacks using type inference on XML databases is proposed.

Key words　database security, XML, inference attack, type inference

1. ま え が き

企業や組織は少なからず機密性の高い情報を保持している．そのような機密情報が格納されているデータベースシステムにおいて，セキュリティ面での管理が必ずしも正確に行われていないことが問題となっている．たとえば，アクセス権の付与が適切に行われておらず，本来権限の無いユーザによる機密情報へのアクセスを許可してしまっている場合がある．また，機密情報への直接のアクセスを許可していない場合でも，アクセスを許可されている情報やドメイン知識などを基に，本来権限の無いユーザがその機密情報を推論できてしまう場合がある [6]．そのような推論を行うことを推論攻撃と呼ぶ．推論攻撃によりどのような情報が得られるかは，一般に自明ではないため，データベース管理者としては，機密情報に対して推論攻撃が成功する可能性をあらかじめ把握できることが重要である．

［例 1］ XML データベースにおける推論攻撃の例を示す．D は学生名とその成績の対応を表す XML 文書であり，以下のスキーマにしたがっているとする．

科目 → 学生*
学生 → 氏名, 点数
氏名 → PCDATA
点数 → PCDATA

すなわち，科目要素の子には学生要素が 0 個以上並び，各学生要素の子には氏名要素と点数要素が 1 つずつ並ぶ．氏名要素と点数要素は文字列 (PCDATA) を値としてとる．

T_1 を，もとの文書における出現順序を保存したまま，科

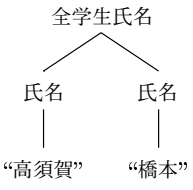

図 1　$T_1(D)$

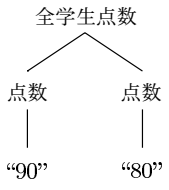

図 2　$T_2(D)$

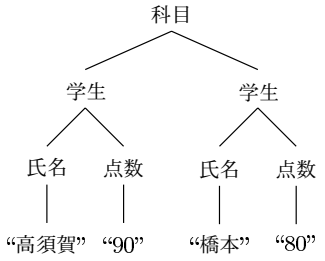

図 3　D

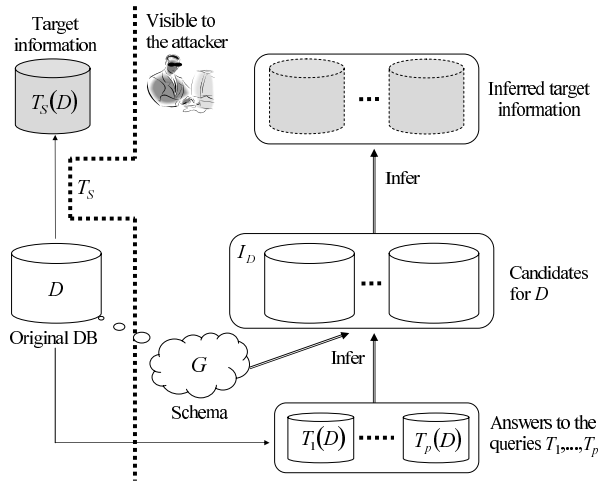

図 4　型推論を用いた攻撃の概略

目の子の学生の子の氏名だけを取り出す問い合わせとする．また，T_2 を，もとの文書における出現順序を保存したまま，科目の子の学生の子の点数だけを取り出す問い合わせとする．

今，T_1 および T_2 の実行が許可されており，実行結果 $T_1(D)$，$T_2(D)$ がそれぞれ図 1，2 に示すとおりであったとする．T_1 の定義および $T_1(D)$ の値より，D は科目の子の学生の子の氏名として “高須賀” という値の要素と “橋本” という値の要素をその順にもつということがわかる．同様に，T_2 の定義および $T_2(D)$ の値より，D は科目の子の学生の子の点数として “90” という値の要素と “80” という値の要素をその順にもつということがわかる．そして，D が上のスキーマにしたがっているということより，D は図 3 に示す文書しかありえないと結論できる．　□

上の例の推論攻撃において問合せとその結果から D の候補を絞り込んでいるが，そこでは型推論が利用できる．型推論とは，問合せとそれへの入力文書の型が与えられたときに，出力文書の型を求める (あるいは，出力文書の型が与えられたときに入力文書の型を求める) ことである．

本稿で扱う，型推論を用いた攻撃の概略を示す (図 4 参照)．ユーザは XML 文書 D のある部分 $T_S(D)$ (ここで T_S は問合せ) の値を知りたい．しかし，ユーザは D 全体を直接見ることができない．ユーザが知ることができるのは，実行を許可されている問合せ $T_1,\ldots,T_p$ の定義と，これらの問合せの D における実行結果 $T_1(D),\ldots,T_p(D)$，および D がしたがうスキーマ G である．このとき，各 i $(1 \leqq i \leqq p)$ について，ユーザは T_i と $T_i(D)$ とから型推論を用いて D の候補を絞り込むことができる．すべての i について D の候補となっている文書のうち，G にしたがう文書から成る集合が，D の候補集合 I_D である．そして，どの候補 $D' \in I_D$ に対しても $T_S(D')$ が同じ値になるのであれば，ユーザは $T_S(D)$ の値を特定できた (すなわち，型推論を用いた攻撃に成功した) ことになる．一方，いくつかの D' について $T_S(D')$ が異なる値になる場合，以下のような両極端な状況が存在しうる．

- $T_S(D')$ の値が数種類の値にしかならない (特定されないまでも攻撃の効果が非常に高い)
- $T_S(D')$ の値が D' ごとに異なる (攻撃の効果がない，あるいは非常に低い)

実用上，これら両極端な状況は区別されるべきケースが多いと考えられるが，本稿ではモデル化の簡潔さのため，いずれの状況についても，“データベースが型推論を用いた攻撃に対して安全である” と定義することにする．

本稿では，まず，型推論を用いた攻撃に対する安全性を形式的に定義する．問合せのクラスとしては，文献 [3] で提案されているトップダウン木変換器を用いる．この木変換器は XSLT の部分クラスに相当する．次に，型推論を用いた攻撃に対する安全性を検証する手法を与える．本検証法の時間計算量は，推論に用いる問合せの個数を定数とみなすと，決定性多項式時間である．

関連研究として，文献 [2] では，各ユーザに security view を提供するというアプローチを提案している．security view は，ユーザがアクセス権をもつ情報をちょうど含んだ文書と，その文書のスキーマ (ビュースキーマ) とから成る．問合せ言語としては XPath を前提としている．元のスキーマからのビュースキーマの生成に関しては，推論攻撃の手がかりとなりうる要素名をつけかえたり削除したりするアルゴリズムを提案している．しかし，どのような推論を行う攻撃者を想定しているかが不明確であり，したがって生成されたビュースキーマの安全性がどのような環境で保証さ

れるのかが不明である．一方，文献 [7] は，推論攻撃を受けたとしても情報の漏洩無く公開できる部分文書を算出するアルゴリズムを提案している．攻撃者が推論に用いることができる情報は，スキーマ情報と関数従属性に関する情報である．本稿では関数従属性に関する情報を扱ってはいないが, [7] では考慮していない型推論に基づく攻撃を扱うことができる．

2. 準　　備

木，生垣，トップダウン木オートマトン，トップダウン木変換器に関する諸定義を行う．そして，型推論と逆型推論について述べる．

2.1 木，生垣

あるアルファベット Σ について，Σ 上の (ランクなし) 木の集合 $\mathcal{T}_\Sigma$ と生垣の集合 $\mathcal{H}_\Sigma$ を，以下を満たす最小集合と定義する:

- $\mathcal{H}_\Sigma = \mathcal{T}_\Sigma^*$,
- $a \in \Sigma$, $h \in \mathcal{H}_\Sigma$ であるならば, $a(h) \in \mathcal{T}_\Sigma$.

ここで $\mathcal{T}_\Sigma^*$ は $\mathcal{T}_\Sigma$ の Kleene 閉包を表す．以下，空生垣 (木の空系列) を ϵ で表す．木において，空生垣を子としてもつ頂点をその木の葉頂点と呼ぶ．木 t(あるいは生垣 h) の大きさ $|t|$(あるいは $|h|$) はその木 (あるいは生垣) の頂点の数である．

2.2 トップダウン木オートマトン

[定義 1] トップダウン木オートマトンは次の 4 つ組 $\mathcal{A} = (Q, \Sigma, q_0, R)$ である:

- Q: 状態の有限集合,
- Σ: アルファベット,
- $q_0 \in Q$: 初期状態,
- R: 遷移規則の集合．ただし，$q \in Q$，$a \in \Sigma$ とし，e を Q 上の正規表現とすると，遷移規則は $(q, a) \to e$ の形式である． □

正規表現 e が表す文字列言語を $L(e)$ と書く．以下，木オートマトン $\mathcal{A} = (Q, \Sigma, q_0, R)$ の動作について述べる．$t = a(\sigma_1(h_1) \cdots \sigma_n(h_n))$ とする．ここで, $\sigma_1, \ldots, \sigma_n \in \Sigma$, $h_1, \ldots, h_n \in \mathcal{H}_\Sigma$ である．t についてその根頂点に状態 q が割り当てられたとき ($q(t)$ と書く), $q_1 \cdots q_n \in L(e)$ であるような $((q, a) \to e) \in R$ が存在するならば, $\mathcal{A}$ は $q(t)$ から $a(q_1(\sigma_1(h_1)) \cdots q_n(\sigma_n(h_n)))$ へ遷移可能であると定義する．$q_0(t)$ から始めて, 以上のような遷移をトップダウンに各部分木について繰り返すことによって, 最終的に t へ遷移可能であるとき, $\mathcal{A}$ は t を受理するという．$\mathcal{A}$ によって受理されるすべての木の集合を $L(\mathcal{A})$ と書く．$\mathcal{A}$ の大きさは $|Q| + |\Sigma| + \sum_{q \in Q, a \in \Sigma} |\mathrm{rhs}(q, a)|$ である．ここで, $\mathrm{rhs}(q, a)$ は $(q, a) \to e$ について e が表す言語を受理する有限オートマトンの大きさである．文字列言語を受理するオートマトンの大きさも, 同様に, 状態数と記号数, そして各遷移規則における遷移先の状態集合の大きさの総和である．

2.3 トップダウン木変換器

Q 中の記号が葉頂点のみに現れるような $\Sigma \cup Q$ 上の木集合を $\mathcal{T}_\Sigma(Q)$ と書く．$\mathcal{H}_\Sigma(Q)$ も同様に定義する．

[定義 2] トップダウン木変換器は次の 4 つ組 $T = (Q, \Sigma, q_0, R)$ である:

- Q: 状態の有限集合,
- Σ: アルファベット,
- $q_0 \in Q$: 初期状態,
- R: 変換規則の有限集合．ただし，$q \in Q, a \in \Sigma \cup \{\lambda\}$, $h \in \mathcal{H}_\Sigma(Q)$ とすると，変換規則は $(q, a) \to h$ の形式である．また，$q = q_0$ のとき h は $T_\Sigma(Q) \setminus Q$ の要素でなければならない． □

トップダウン木変換器 $T = (Q, \Sigma, q_0, R)$ の動作について述べる．木 $t = a(t_1 \cdots t_n)$ は, 状態 $q \in Q$ で T によって次のような $T^q(t)$ へ変換される:

- ある h に対して $((q, \lambda) \to h) \in R$ ならば, $T^q(t)$ は, h 中のすべての状態記号 p を $T^p(t)$ に置換して得られる生垣と等しい．
- ある h に対して $((q, a) \to h) \in R$(ただし, $a \in \Sigma$) ならば, $T^q(t)$ は，h 中のすべての状態記号 p を生垣 $T^p(t_1) \cdots T^p(t_n)$ に置換して得られる生垣と等しい．特に $t = a(\epsilon)$ のときは, $T^q(t)$ は, h 中のすべての状態記号 p を ϵ に置換して得られる生垣と等しい．
- 左辺に (q, a) をもつ変換規則が R に存在しないならば, $T^q(t) = \epsilon$ である．

t の T による変換 $T(t)$ を $T(t) = T^{q_0}(t)$ と定義する．T の大きさは $|Q| + |\Sigma| + \sum_{q \in Q, a \in \Sigma} |\mathrm{rhs}(q, a)|$ である．ここで, $|\mathrm{rhs}(q, a)|$ は $((q, a) \to h) \in R$ について $|h|$ である．

同じ左辺を持つ規則が高々1 つであるとき，変換器は決定性であるという．初期状態に関する変換規則の制約によりこの変換器の出力が木であることが保証される．また変換規則の中で，右辺に状態が 2 つ以上存在する規則は，規則を適用している頂点以下の部分木をコピーするので copying rule と呼び，右辺にアルファベットを含まない規則は，規則を適用している頂点を除去する変換を行うので deletion rule と呼ぶ．そして，copying rule を持たない木変換器を non-copying，deletion rule を持たない木変換器を non-deleting と呼ぶ．

2.4 型 推 論

木オートマトンは XML スキーマの形式的モデルの 1 つであり，XML 文書の “型” を表現するのに用いられる．ある木変換器 T と木オートマトン $\mathcal{A}_{in}$ について，$L(\mathcal{A}'_{out}) = \{T(t) \mid t \in L(\mathcal{A}_{in})\}$ であるような木オートマトン $\mathcal{A}'_{out}$ を求めることを型推論という．逆

に，ある木変換器 T と木オートマトン $\mathcal{A}_{out}$ について，$L(\mathcal{A}'_{in}) = \{t \mid T(t) \in L(\mathcal{A}_{out})\}$ であるような木オートマトン $\mathcal{A}'_{in}$ を求めることを逆型推論という．本稿では逆型推論も単に型推論と呼ぶ．

3. 型推論を用いた攻撃とそれに対する安全性

本節では，型推論を用いた攻撃とそれに対する安全性を定義する (図 5 参照)．

ユーザ (攻撃者) が XML 文書 D への攻撃に用いることができる情報は以下の通りである．

- $T_1, T_2, \ldots, T_p$: ユーザが実行を許可されている問合せ．
- $T_i(D)$: D に対する問合せ T_i の実行結果．
- D のスキーマ (を表す木オートマトン) $\mathcal{A}_G$．
- T_S: D 中の機密情報を返す問合せ．

以下の集合

$$I_S = \{T_S(D') \mid D' \in L(\mathcal{A}_G), \\ T_1(D') = T_1(D), \ldots, T_p(D') = T_p(D)\}$$

が要素を 1 つしかもたないとき，$T_S(D)$ は型推論を用いた攻撃に対して安全でないという．そうでないとき，$T_S(D)$ は安全であるという．

上の集合を定義どおり求めるためには，まず，T_i の実行結果と $\mathcal{A}_G$ を用いて D' が動く範囲 I_D を決定しなければならない．これは，T_i に関する型推論の結果とスキーマ $\mathcal{A}_G$ の情報により求めることができる．本稿では，T_i として，non-copying かつ non-deleting な木変換器 T_{i1} と non-copying な木変換器 T_{i2} の合成により定義されている問合せを考える．T_{i1} は，コピーや削除以外の変換を行いつつ，後に T_{i2} によって削除されるべき頂点のラベルを ♯ に変換する機能を持つ．そして T_{i2} は ♯ とラベル付けされた頂点を削除するだけの機能を持つ木変換器である (図 5 参照)．このような T_i に対して型推論を行う (したがって D' が動く範囲 I_D を求める) 方法を次節で示す．

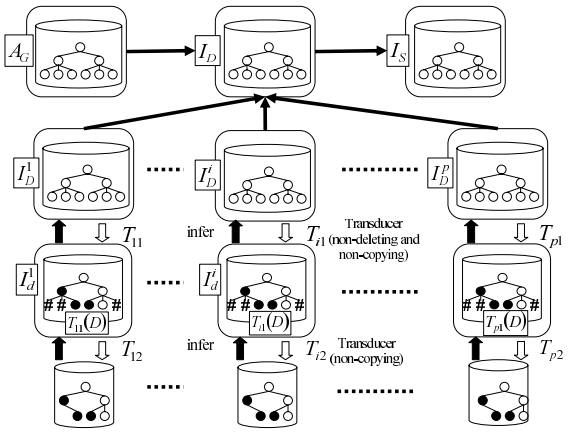

図 5　実行結果 $T_i(D)$ について型推論を用いた攻撃

D' が動く範囲 I_D が求まると，次は，各 $D' \in I_D$ に対して T_S を実行した結果の集合 I_S を求める必要がある．これは T_S に関する型推論に相当する．本稿では，T_S として，non-copying な木変換器により定義されている問合せを考える．そして，型推論を行う方法を次節で示す．

4. 安全性検証

本稿において提案する安全性検証の詳細について述べる (図 5 参照)．ユーザに対して許している問合せ T_i の実行結果 $T_i(D)$ から，$\mathcal{A}_D$ の候補を求める方法を 4.1 で述べる．そしてその候補に対して機密情報を求める問合せ T_S を実行し，データベースインスタンスの機密情報候補がただ 1 つに絞り込まれないか (すなわちデータベースが型推論を用いた攻撃に対して安全であるか) を検証する方法を 4.2 で述べる．

4.1 データベースインスタンスの候補の絞込み

本節では問合せ結果 $T_1(D), \ldots, T_p(D)$ から元のデータベースインスタンス D の候補 I_D を受理する木オートマトン $\mathcal{A}_D$ 求める方法 (図 5 参照) について述べる．$\mathcal{A}_D$ は以下の 3 ステップにより求める．

（1） T_{i2} の出力として $T_i(D)$ が得られるような, T_{i2} への入力木の候補集合 I_D^i を受理するオートマトン $\mathcal{A}_d$ を求める．

（2） T_{i1} の出力として I_d^i 中のいずれかの木が得られるような, T_{i1} への入力木の候補集合，すなわち問合せ結果 $T_i(D)$ から推論される元のデータベースインスタンス D の候補の集合 I_D^i を受理するオートマトン $\mathcal{A}_D^i$ を求める．

（3） すべての $\mathcal{A}_D^i$ の交わり $\mathcal{A'}_D$ を求めて, さらに $\mathcal{A'}_D$ と $\mathcal{A}_G$ との交わり $\mathcal{A}_D$ を求める．

直観的にはステップ (1) において，T_{i2} において削除された ♯ の復元，すなわち削除されたノードの位置として可能なものを全て洗い出し，♯ が削除される前の木の候補を求め，ステップ (2) において，ステップ (1) で求められた木に対し，T_{i1} をもとに元の木 D の候補を求め，そしてステップ (3) で，許可している各問合せについて求められた候補と，スキーマとから元の木 D の候補を絞り込む．

4.1.1 では $T_i(D)$ から T_{i2} の入力の候補 I_d^i を受理するオートマトン $\mathcal{A}_d^i$ を求める方法，4.1.2 では，$\mathcal{A}_d^i$ から T_{i1} の入力の候補 I_D^i を受理するオートマトン $\mathcal{A}_D^i$ 求める方法について述べる．そして 4.1.3 では，すべての $\mathcal{A}_D^i$ と $\mathcal{A}_G$ から D の候補 I_D を受理する $\mathcal{A}_D$ を求める方法について述べる．

またステップ (1)(2) で扱うトップダウン木変換器 T_{i1}, T_{i2} は，変換規則を以下の形式に限定したものとする．

- $(q,a) \to a'(q'')$
- $(q'',\lambda) \to b'(q')$

4.1.1 ステップ (1)

ここでは $T_{i2}(T_{i1}(D)) = T_i(D)$ であるような，$T_{i1}(D)$ を受理する木オートマトン $\mathcal{A}^i_d = (Q^i_d, \Sigma^i_d, q^i_{d0}, R^i_d)$ を構成する．まず $T_i(D)$ について，$T_i(D)$ のみを受理するトップダウン木オートマトン $\mathcal{A}_O = (Q_O, \Sigma_O, q_{O_0}, R_O)$ を構成する．この構成の仕方については省略する．

最初に $\mathcal{A}_O$ の各遷移規則 $r = (q,a) \to e_r \in R_O$ について，正規表現 e_r を表す言語 $L(e_r)$ を受理する有限オートマトン $M_r = (Q_r, N_r, \delta_r, q_r, F_r)$ を考える．ここで $|Q_r| = n_r$，$Q_r = \{q^r_1, q^r_2, \ldots, q^r_{n_r}\}$ とする．また，M_r において，全ての 2 つの状態の対 q^r_i, q^r_j について，q^r_i から q^r_j へ到達する語全体を表す正規表現を e^r_{ij} とする．いま，新たな有限オートマトン $M_{r'} = (Q_r, N_{r'}, \delta_{r'}, q_r, F_r)$ を構成する．ただし，$N_{r'} = N_r \cup \{b^r_{ij} | 1 \leqq i,j \leqq n_r, L(e^r_{ij}) \neq \emptyset\}$，$\delta_{r'} = \delta_r \cup \{(q^r_i, b^r_{ij}, q^r_j) | 1 \leqq i,j \leqq n_r, L(e^r_{ij}) \neq \emptyset\}$ とする．ここで $M_{r'}$ に受理される語全体を現す正規表現を $e^{r'}$，$M_{r'}$ において，全ての 2 つの状態の対 q^r_i, q^r_j について，q^e_i から q^e_j へ到達する語全体を表す正規表現を $e^{r'}_{ij}$ とする．このとき，$\mathcal{A}^i_d$ は以下により構成される．

$$
\begin{aligned}
Q^i_d &= Q_O \cup \bigcup_{r \in R_O} \{b^r_{ij} | 1 \leqq i,j \leqq n_r\} \\
\Sigma^i_d &= \Sigma_O \cup \{\sharp\} \\
q^i_{d0} &= q_{O_0} \\
R^i_d &= R_O \cup \bigcup_{r \in R_O} \{\{(q,a) \to e_{r'}\} \\
&\qquad \cup \{(b^r_{ij}, \sharp) \to e^{r'}_{ij} \mid 1 \leqq i,j \leqq n_r, L(e^r_{ij}) \neq \emptyset\}\Big\}
\end{aligned}
$$

4.1.2 ステップ (2)

木オートマトン $\mathcal{A}^i_d = (Q^i_d, \Sigma^i_d, q^i_{d_0}, R^i_d)$ と $T_{i1} = (Q_{i1}, \Sigma_{i1}, q^0_{i1}, R_{i1})$ から，元のデータベースインスタンス D の候補 I^i_D を受理する木オートマトン $\mathcal{A}^i_D$ を構成する方法を示す．$\mathcal{A}^i_D$ は，入力木を T_{i1} によって変換して，それが $\mathcal{A}^i_d$ によって受理されるかをシミュレートする．

$\mathcal{A}^i_D = (Q_{i1} \times Q^i_d, \Sigma_{i1}, (q^0_{i1}, q^i_{d_0}, R^i_D)$ における遷移規則 R^i_D の構成は以下のように行う．

- R_{i1} 中の変換規則: $(q_{i1}, a) \to a'(q'_{i1})$ と，R^i_d 中の遷移規則 $(q^i_d, a') \to P^i_d$ から，R^i_D に以下の規則を作る．

$$((q_{i1}, q_d), a) \to P'^i_d$$

(P'^i_d: P^i_d を構成する各状態 q'^i_d を (q'_{i1}, q'^i_d) と書き換えたもの)

この $\mathcal{A}^i_D$ によって単一の問合せ実行結果 $T_i(D)$ から推論される D の候補が認識される．

4.1.3 ステップ (3)

ここではステップ (2) までで求められた $\mathcal{A}^i_D$ と D がしたがっているスキーマ G にしたがう XML 文書を認識する木オートマトン $\mathcal{A}_G$ から D の候補 I_D を受理する木オートマトン A_D を求める方法について述べる．

まず，上で得られた $\mathcal{A}^i_D, (i = 1, \ldots, p)$ の交わりをとり，これを $\mathcal{A'}_D$ とする．そして D はスキーマ G に従っているため，$\mathcal{A}_G$ と $\mathcal{A'}_D$ との交わりを求める．ただしこのとき，実際のデータ (値) をアルファベットに含んでいる $\mathcal{A'}_D$ に対し $\mathcal{A}_G$ はスキーマを表現しているため，$\mathcal{A'}_D$ における実際の値に対応するアルファベットは PCDATA のようなデータ型となってしまう．このため $\mathcal{A}_G$ と $\mathcal{A'}_D$ との交わりを厳密にとると，D を受理しないような木オートマトンになってしまうため，$\mathcal{A'}_D$ の中で実際のデータを表す記号の集合と $\mathcal{A}_G$ の中でそのデータの型に合う記号の集合とは同じものとみなすという条件を置いたうえで交わりを求めることとする．

これにより得られるものが D の候補 I_D を受理する木オートマトン $\mathcal{A}_D$ である．

4.2 機密情報の推論

データベースインスタンス D 中の機密情報 $T_S(D)$ の候補の求め方について述べる．4.1.3 までで求められている $\mathcal{A}_D$ は D の候補を受理する木オートマトンであり，I_D に対して木変換器 T_S による変換を行った結果は，データベースインスタンス D 中の機密情報 $T_S(D)$ の候補である．ここでは木オートマトン $\mathcal{A}_D = (Q_D, \Sigma_D, q_{D_0}, R_D)$ と $T_S = (Q_S, \Sigma_S, q^0_S, R_S)$ から，$T_S(D)$ の候補 I_S) を受理する木オートマトン $\mathcal{A}_S$ を構成する方法を以下に述べる．

ここで T_S は，変換規則を以下の形式に限定したものとする．

- $(q,a) \to a'(q'')$

$\mathcal{A}_S = (Q_S \times Q_D, \Sigma_S, (q^0_S, q_{D_0}), R^S)$ における遷移規則 R^S は次のように構成する．

- R_S 中の変換規則 $(q_S, a) \to a'(q'_S)$ と R_D 中の遷移規則 $(q_D, a) \to P_D$ から，R^S に以下の規則を作る．

$$((q_S, q_D), a') \to P'_D$$

(P'_D: P_D を構成する各状態 q'_D を (q'_S, q'_D) と書き換えたもの)

以上より求められた木オートマトン $\mathcal{A}_S$ に受理される木の集合の要素が 1 つでないとき，このデータベースインスタンス D 中の機密情報 $T_S(D)$ が一意に求まらないことから，データベースが型推論を用いた攻撃に対して安全であるといえる．これを検証する方法として以下のような方法がある．まず $\mathcal{A}_S$ はランクなしの木を入力とするトップダウン木オートマトンであり，これを等価なボトムアップの (ランクなし) 木オートマトンに書き換える．トップダウンの木オートマトンを等価なボトムアップの木オートマトンに書き換えるには，トップダウン木オートマトンの初期状

態をボトムアップ木オートマトンの受理状態にし，各遷移規則の左辺と右辺を入れかえればよい．次にこの木オートマトンを等価なボトムアップの二分木オートマトン (binary tree automaton) に書き換える．このボトムアップ二分木オートマトンが受理する木の集合が要素をただ 1 つだけ持つかどうかを判定する方法 [1] を用い，$\mathcal{A}_S$ が受理する木の集合が要素をただ 1 つだけ持つかどうかを判定する．

これにより機密情報 $T_S(D)$ が型推論を用いた攻撃に対して安全であるかどうかを判定する．

5. 時間計算量の評価

本節では本検証法の時間計算量について述べる．

まずステップ (1) の計算 (4.1.1 参照) については，この計算量は $|T_i(D)|$ についての LOGSPACE であることが知られている [4]．またここで求められる $\mathcal{A}_d^i$ の大きさは，$|T_i(D)|$ の多項式の大きさとなる．

次に，ステップ (2) の計算 (4.1.2 参照) は，non-copying かつ non-deleting の木変換器 T_{i1} の各変換規則につき，その右辺の記号を左辺に持つ $\mathcal{A}_d^i$ の遷移規則を選び，それらから新たに遷移規則を作るという作業を繰り返すことから，時間計算量は $O(|T_{i1}| \times |\mathcal{A}_d^i|)$ とできる．

ステップ (3) の計算 (4.1.3 参照) の時間計算量は，$O(|\mathcal{A}_D^1| \times \cdots \times |\mathcal{A}_D^p| \times |\mathcal{A}_G|)$ となる．p を定数とすると多項式時間となる．$|\mathcal{A}_D|$ についても，状態集合の大きさが各オートマトンの状態集合の大きさの積となることが考えられるので，これも時間計算量と同様，p を定数とすると多項式の大きさとなる．

$\mathcal{A}_D$ と T_S から $\mathcal{A}_S$ を求める計算 (4.2 参照) は，$\mathcal{A}_D^i$ を求める計算と同様，各変換規則と遷移規則から新たな遷移規則を作るという作業になるため，この時間計算量は $O(|\mathcal{A}_D| \times |T_S|)$ となる．

最後に $\mathcal{A}_S$ が受理する木の集合 I_S の要素がただ 1 つであるかどうかを検証する計算 (4.2 参照) について考える．まずトップダウンの木オートマトンを等価なボトムアップの木オートマトンに書き換える時間計算量は木オートマトンのサイズについて線形時間である．次に非決定性の (ボトムアップランクなし) 木オートマトンから，二分木オートマトンへの変換にかかる実行時間は多項式時間であることがわかっている [5]．またボトムアップ二分木 (ランクあり) オートマトンが受理する木の集合が要素をただ 1 つだけ持つかどうかの判定も時間計算量は多項式時間であることが知られいる [1]．よってこの計算は $|\mathcal{A}_S|$ についての多項式時間である．

以上より，本検証法の時間計算量は $\mathcal{A}_D$ を求める際に交わりを求める木オートマトンの数，すなわち推論に用いる問合せの個数を定数とみなしたとき，多項式時間である．

6. あとがき

本稿では XML データベースへの型推論を用いた攻撃を形式的に定義し，それに対する安全性を検証する手法を提案した．本検証法の時間計算量は，攻撃者が推論に用いることのできる問合せの個数を定数とみなしたとき，決定性多項式時間である．

今後の課題としては，まず，対象とする問合せクラスの拡張が挙げられる．現在問合せのモデルとして用いているトップダウン木変換器では，変換規則への制限が多い．まず変換規則の右辺に生垣や，枝分かれのある木を許していないため，問合せに対して新たな追加情報を加えて返すような動作ができない．そこで，この制限を緩めることについて検討をはじめている．また，たとえば「A という子要素をもつときかつそのときのみ，親要素を削除する」ような動作ができない．そこで，子や孫への先読み機能をもつ木変換器に拡張することについて検討をはじめている．次に，推論に用いることのできる情報として，[7] のような関数従属性を考慮する必要があると考えられる．また，安全性の定義に関わる課題として，1. 節で述べた「両極端な状況」の間に安全かそうでないかの線引きを行うことがあげられる．また，本稿は XML 文書 D が具体的に与えられている場合での安全性検証を扱っているが，静的な安全性検証，すなわち，$T_S(D)$ が安全でないような D が存在するか否かを検証する問題も，実用上極めて重要な課題である．

文　献

[1] H. Comon, M. Dauchet, R. Gilleron, F. Jacquemard, D.Lugiez, S. Tison, and M. Tommasi: “Tree Automata Techniques and Applications,” http://www.grappa.univ-lille3.fr/tata/.

[2] W. Fan, C.-Y. Chan, and M. Garofalakis: “Secure XML Querying with Security Views,” Proceedings of the 23rd SIGMOD International Conference on Management of Data, pp. 587-598, 2004.

[3] W. Martens and F. Neven: “Frontiers of Tractability for Typechecking Simple XML Transformations,” Proceedings of the 23rd ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems, pp. 23-34, 2004.

[4] W. Martens and F. Neven: “On the Complexity of Typechecking Top-Down XML Transformations,” Theoretical Computer Science, Volume 336, Issue 1, pp. 153-180, 2005.

[5] F. Neven: “Automata theory for XML researchers,” ACM SIGMOD Record Volume 31, Issue 3, pp. 39-46, 2002.

[6] C. P. Pfleeger and S. L. Pfleeger: “Security in Computing,” 3rd Ed., Prentice Hall, 2002.

[7] X. Yang and C. Li: “Secure XML Publishing without Information Leakage in the Presence of Data Inference,” Proceedings of the 30th VLDB Conference, pp. 96-107, 2004.