동적 XML 데이타 스트림의 안전한 질의 처리를 위한 효율적인 접근제어 레이블링

(Efficient Access Control Labeling for Secure Query Processing on Dynamic XML Data Streams)

안 동 찬 [†] 박 석 [††]
(Dong Chan An) (Seog Park)

요 약 유비쿼터스 데이타 스트림 환경에서 동적 XML 데이타 스트림에 대한 효율적이고 안전한 접근제어 기법은 최근에 활발한 연구분야가 되어왔다. 본 논문에서 동적 XML 데이타 스트림 환경에서 효율적이고 안전한 접근제어를 위한 향상된 롤-기반 소수 레이블링 기법을 제안한다. 또한 지속적으로 갱신되는 XML 문서의 효율적인 레이블링 기법과 효율적이고 안전한 접근제어를 위한 기존연구의 문제점을 지적한다. 제안하는 향상된 레이블링 기법은 문서가 갱신될 때 다시 레이블링 하지 않고도 노드 사이의 조상-후손 관계와 형제 관계를 유지할 수 있으며, 레이블의 충돌 없이 XML 트리에 임의의 지속적인 노드 삽입 또는 갱신을 지원한다. 또한, 롤-기반 소수 레이블링 기법을 통해 효율적인 접근제어를 구현하였다. 끝으로 본 논문의 제안 방법이 효율적이고 안전함을 실험을 통해 보여줄 것이다.

키워드 : 동적 XML 데이타, 접근제어, 질의처리, 데이타 스트림, 롤-기반 소수 레이블링

Abstract *Recently, the needs for an efficient and secure access control method of dynamic XML data in a ubiquitous data streams environment have become an active research area. In this paper, we proposed an improved role-based prime number labeling scheme for an efficient and secure access control labeling method in dynamic XML data streams. And we point out the limitations of existing access control and labeling schemes for XML data assuming that documents are frequently updated. The improved labeling method where labels are encoded ancestor-descendant and sibling relationships between nodes but need not to be regenerated when the document is updated. Our improved role-based prime number labeling scheme supports an infinite number of updates and guarantees the arbitrary nodes insertion at arbitrary position of the XML tree without label collisions. Also we implemented an efficient access control using a role-based prime number labeling. Finally, we have shown that our approach is an efficient and secure through experiments.*

Key words : Dynamic XML Data, Access Control, Query Processing, Data Streams, Role-based Prime Number Labeling

· 본 연구는 한국과학재단 세계수준의 연구중심대학(WCU) 육성사업(R33-2008-000-10110-0) 지원으로 수행되었음
· 이 논문은 제35회 추계학술발표회에서 '동적 XML 데이터 스트림의 질의 처리를 위한 효율적인 접근제어 레이블링 기법'의 제목으로 발표된 논문을 확장한 것임

† 학생회원 : 서강대학교 컴퓨터공학과
channy@sogang.ac.kr
†† 종신회원 : 서강대학교 컴퓨터공학과 교수
spark@sogang.ac.kr

논문접수 : 2008년 12월 19일
심사완료 : 2009년 2월 11일

정보과학회논문지: 데이타베이스 제36권 제3호(2009.6)

1. 서 론

XML은 인터넷 상에서 데이타의 교환 및 표현의 일반적인 표준으로 널리 사용되고 있다. XPath[1]와 XQuery[2] 같은 질의어는 XML 데이타를 목적으로 W3C 그룹에서 개발되었으며, XPath와 XQuery의 효율적이고 안전한 처리는 매우 중요한 이슈이기도 하다. XML은 논리적으로 부모-자식 관계 또는 조상-후손 관계처럼 노드 사이에 트리 구조를 이루고 있다. 기존 연구에서 XML 문서의 접근제어를 위해 구조 인덱싱과 노드 레이블링 방법과 같은 많은 제안이 있었으나, 상대적으로 질의 접근제어를 위한 XML 데이타 접근제어의

실행에 대한 연구는 부족하였다. 더욱이, 기존 접근제어 환경은 유한하고, 영속적이며, 정적인 시스템 중심의 환경이었다면, 최근 접근제어는 무한하고, 일시적이며, 동적이고 연속적인 데이타 스트림 환경에 대한 요구가 증가되고 있다[3]. 이와 같이 기존 접근제어 모델은 데이타 스트림 환경에 적용하기 어려운 점이 있다[4].

또한, 기존 연구[5-7]들이 XML 문서의 갱신시 순서를 유지하였으나 역시 최대의 단점은 삽입 또는 갱신시 저장공간과 같은 많은 레이블링 비용이 든다는 것이며[8], XML 문서의 보안은 전혀 고려되지 않았다.

따라서, 본 논문에서 동적 XML 데이타 스트림 환경에서 효율적이고 안전한 질의 처리를 위한 향상된 방법을 제안한다. 제안한 방법은 자원 제약적 환경인 모바일 단말기에서 효율적이고 안전한 실시간 질의 처리가 가능하고 충돌발생이나 다시 레이블링 하지 않고 XML 트리의 임의의 노드에 연속적인 삽입을 가능하게 해준다. 제안하는 레이블링 방법을 통해 동적 환경에서도 XML 문서의 효율적인 관리가 가능하며, 레이블링의 최종 목적인 안전한 질의 처리도 가능하게 해 준다. 안전한 질의 처리는 XML 문서의 레이블링 즉, 롤-기반 소수 레이블링 방법을 통해 소수의 특성을 활용하여 빠르고 안전한 접근제어를 구현하였다.

본 논문은 다음과 같이 이루어져 있다. 이어지는 제2장에서 XML의 접근제어와 레이블링 관련 연구를 살펴보고, 제3장에서 제안한 방법에 대해 소개하며, 제4장에서 본 논문의 제안 방법이 효율적임을 실험을 통해 입증하며, 끝으로 제5장에서 결론으로 본 논문을 마무리한다.

2. 관련연구

전통적인 XML 접근제어 이행 메커니즘[9-12]은 뷰-기반 이행 메커니즘 이었다. 사용자에 대한 접근제어의 의미는 접근제어 정책에 따라 특정 뷰를 제공하는 것이다. 이 방법은 트리 레이블링을 통해 뷰를 계산하는 유용한 알고리즘을 제공한다. 그러나, 뷰-기반 이행 메커니즘은 사용자의 수가 증가 될수록 알고리즘이 스케일러블하지 못하며, 고비용이 요구된다[13].

뷰-기반 이행 메커니즘의 문제를 극복하기 위해, M. Murata[13] 등은 접근제어 정책에 위배되는 질의를 제거하는 필터링 방법을 제안했으며, B. Luo[14] 등은 관련 접근제어의 조합을 이용한 질의 재작성을 통해 질의 처리를 시도하였다. 그러나 접근제어 규칙의 공유 비결정적 오토마타(NFA)는 사용자에 의해 생성되나, 공유 NFA는 사용자의 질의 관점에서 생성된 많은 불필요한 접근제어 규칙을 갖게 되고, 사용자의 질의에 대한 수용, 거절 또는 재작성의 결정을 위한 시간을 필요로 하게 된다.

XML 데이타 모델과 같은 계층적 데이타 모델에서는 보안 관리자에 의해 정의된 권한부여를 '명시적(explicit)' 이라 하며, 명시적 권한부여에 기초하여 시스템이 파생한 권한부여를 '묵시적(implicit)'이라 한다. 또한, 저장공간의 장점을 살리기 위해 묵시적 권한부여 방법을 사용하면서 적절히 '전파정책(propagation policy)'을 만든다. 서로 다른 환경에서 최적화된 전파정책을 가정하여 '가장 특정화된 것을 우선으로 하는 정책(most-specific-precedence)'이 일반적으로 사용된다. 이와 같은 묵시적 권한부여에 의한 전파정책에 의해 발생될 수 있는 '충돌(conflict)' 문제를 해결하는 정책으로는 '거절우선정책(denial-takes-precedence)'을 일반적으로 사용한다. 또한 긍정적(positive) 권한부여와 부정적(negative) 권한부여를 혼합하여 사용하기 때문에 명시적인 권한부여가 없는 노드에 대한 '결정정책(decision policy)'으로 명시적 권한부여가 없는 노드는 접근을 불허하는 '폐쇄정책(closed policy)'을 사용한다[13].

제한된 자원을 가진 이동 단말기 등에서 대용량의 XML 데이타를 효율적으로 관리하기 위해 XML 데이타를 적당한 크기의 조각(fragmentation)으로 나누어 전송하여 질의 처리하는 방법이 연구되었다[11,15]. 이들 연구는 센서네트워크 환경에서 XML 데이타가 계속해서 생성되고, 이동단말기의 메모리 효율과 질의처리시간을 고려하여, 구조적으로 분할하여 전송되는 XML 조각 스트림(streams)을 의미한다. 더욱이 XML 데이타 스트림 환경에서 특정 데이타가 갱신될 때, XML 문서 전체가 아니라 변경된 조각 데이타만 갱신을 하므로 전송비용을 절감할 수 있다. Hole-Filler 모델[4,10]에서는 XML 데이타를 구조적으로 분할 하는 방법을 제안하였다. XFrag[16]와 XFPro[17]에서는 Hole-Filler 모델을 적용하여 XML 조각 데이타 처리를 제안하였다. 그럼에도 불구하고, 이 방법들은 Hole-Filler 모델의 추가적인 정보 공간의 요구와 늦은 처리 시간의 문제점을 가지고 있다. XFPro는 파이프라인을 이용하여 처리 시간을 향상시켰으나, Hole-Filler 모델의 근본적인 문제는 해결하지 못하였다. 그림 1은 의료정보XML문서의 예[18]이고, 그림 2는 Hole-Filler 모델[16]에 의해 분할된 XML 문서의 예이다.

엘리먼트의 삽입과 삭제가 용이한 동적 XML 문서의 질의처리를 위해 XML 레이블링 기법이 많이 사용된다. 기존의 레이블링 기법은 잦은 갱신에 부적합하고, 검색을 위해 모든 노드의 레이블을 재탐색 해야하는 문제점이 있다[13,19].

X. Wu[20] 등의 레이블링 기법은 동적 XML 문서를 위해 제안된 레이블링 기법이다. 이 기법은 기존 노드의

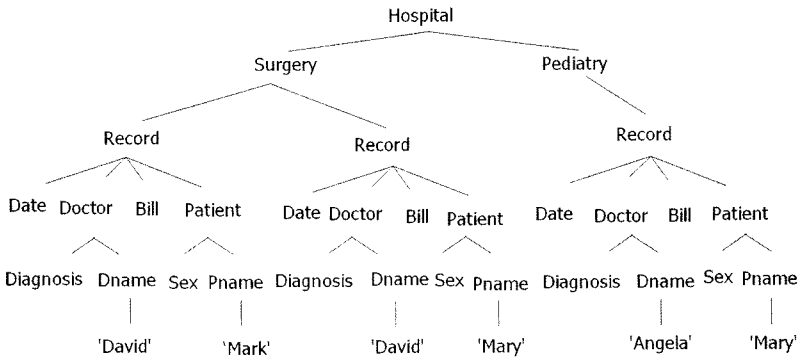

그림 1 의료정보 XML 문서

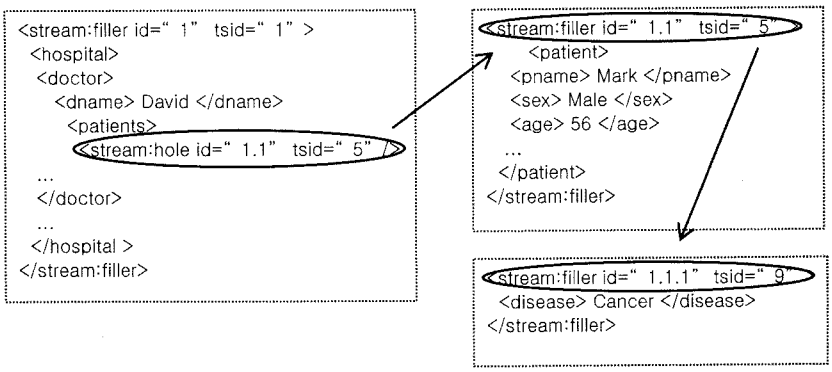

그림 2 홀-필러 모델(Hole-Filler Model)에 의해 단편화된 XML 문서

레이블에 거의 영향을 주지 않고 레이블이 가능한 고유의 소수(prime number) 레이블링 기법이다. 각각의 노드에 소수를 할당하는데 이것은 조상-후손 관계를 유지하며, 문서가 갱신될 때 다른 노드에 영향을 주지 않도록 설계되었다. 그러나 갱신과정 동안 갱신 순서 정보에 대한 고려를 위해 XML 문서의 재탐색 과정에 대한 비용은 여전히 존재한다. 본 논문에서는 소수 레이블링과 달리 롤-기반 소수 레이블링을 사용하여, 제한된 롤에 소수를 할당하여 접근제어 여부를 판단하는 방법을 사용한다. 그리고 갱신 등의 변경에 대한 노드 관계는 숫자와 문자를 이용한 레이블링을 통해 충분히 판단할 수 있다.

3. 질의 처리

동적 롤-기반 소수 레이블링 기법(dynamic role-based prime number labeling scheme)의 제안 환경은 그림 3과 같다. 의료정보에 대한 질의요청은 접근제어 정책에 따라 허가된 사용자이고, 허가된 사용자에 대한 응답은 적절한 시간 내에 이루어져야 한다.

3.1 접근제어 레이블링 기법

동적 롤-기반 소수 레이블링 기법은 그림 3과 같은 환경에서 설명된다. 먼저, 제안하는 환경의 특징을 고려하면 그림 1에서 XML 문서의 조각이 그림 4와 같이 조각으로 나뉘어진다. 기존 연구 XFrag[16]에서 추가적인 정보공간의 요구와 낮은 질의 처리 시간과 같은 문제들을 최소화 한 것이 그림 4에서 보여진다. 이것은 태그 스트럭처(tsid)와 같은 추가적인 정보 즉, XML 문서의 순서정보가 고려대상이 되지 않고 필요 없음을 의미한다.

XML 데이타 스트림의 조각 분할과정을 거친 후 표 1의 롤-기반 소수를 이용하여 의료정보 XML 문서의 각 노드에 적절한 동적 롤-기반 소수의 곱이 할당 된다. 여기서 소수의 곱은 해당 노드에 접근 가능한 롤들의 곱을 의미한다. 어떤 조직 내에서 롤의 수에는 한정적

표 1 롤-기반 소수 테이블

Roles	Role-Based Prime Number
Patient	2
Doctor	3
Researcher	5
Insurer	7

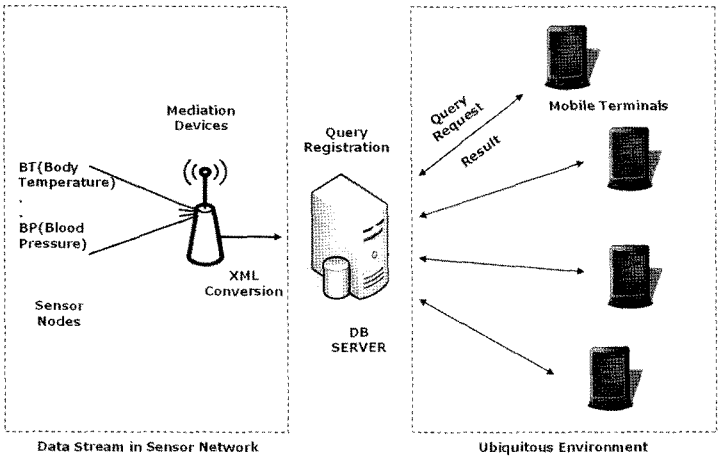

그림 3 XML 데이타 스트림 환경에서 질의처리 과정

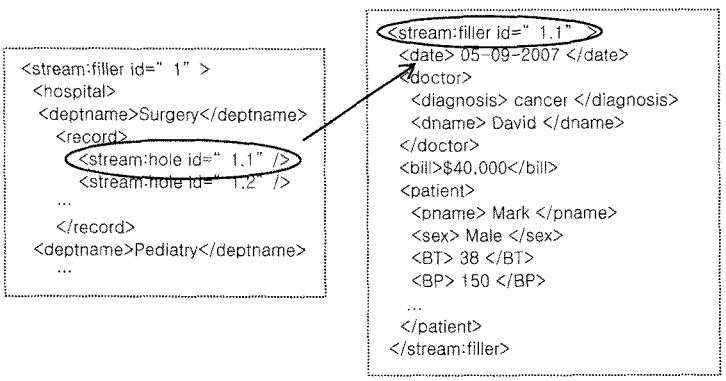

그림 4 개선된 XML 단편화 데이타 스트림

이므로 소수로 표현하는 것에는 문제가 없다.

레이블링 표기. 모든 노드의 레이블은 네 개(l, L1, L2, 그리고 L3)의 구성요소로 이루어진다.

l L1.L2.L3

1. 레벨(l) - XML 문서 내의 노드 레벨을 의미한다. 트리의 루트(root) 노드에서 리프(leaf) 노드까지 부여된다. 그러나 루트 노드의 레이블은 널(null)이다.
2. 조상 레이블(L1) - 부모 노드의 레이블을 상속받는다. 단, 부모 노드의 레벨은 제외된다.
3. 자기 레이블(L2) - 자기 자신의 레이블을 의미한다.
4. 롤-기반 소수 곱(L3) - 해당 노드에 접근 가능한 롤-기반 소수들의 곱을 의미한다.

각 노드의 고유한 레이블은 네 개의 구성요소로 이루어지며, 한정자(.)에 의해 표기된다.

레이블링 기법. XML 문서의 레이블링은 아래와 같다.

1. 루트 노드는 항상 첫 번째 레벨이다. 형제(sibling) 노드도 부모(parent) 노드도 없으므로 루트 노드의 레이블은 널(null)이다.
2. 루트 노드의 첫 번째 자식(child) 노드(N1)는 "1a1.L3"이고, 루트 노드의 두 번째 자식(child) 노드(N2)는 "1b1.L3"이다. 왜냐하면, 부모 노드의 레이블은 널이고 l과 L2가 병합된다. 세번째 성분 L3는 이번 레벨에서는 선택사항이다. 만약, 모든 롤이 접근 가능하면 L3는 생략한다.
3. 두 번째 레벨 노드 N1의 첫 번째 자식 노드(NN1)는 "2a1.a1.L3"이고, 두 번째 레벨 노드 N2의 첫 번째 자식 노드(NN2)는 "2b1.a1.L3"이다. 부모 노드의 레이블(L1)은 상속된다. 세번째 성분 L3는 노드의 접근 가능한 롤-기반 소수들의 곱이다.
4. 세 번째 레벨 노드 NN1의 첫 번째 자식 노드(NNN1)는 "3a1a1.a1.L3"이고, 세 번째 레벨 노드 NN2의 첫 번째 자식 노드(NNN2)는 "3b1a1.a1.L3"이다. 여기서 부모 노드의 레이블 L1, L2가 상속된다.

5. 네 번째 레벨 노드 NNN1의 첫 번째 자식 노드(NNNN1)는 "4a1a1a1.a1.L3"이고, 네 번째 레벨 노드 NNN2의 첫 번째 자식 노드(NNNN2)는 "4b1a1a1.a1.L3". 마찬가지로 부모 노드의 레이블 L1, L2가 상속된다.
6. 세 번째 구성 성분(L3)은 아래와 같이 해당 노드에 접근 가능한 롤-기반 소수들의 곱으로 이루어진다.
   - Date (210) : 2, 3, 5, 7의 곱
   - Doctor (30) : 2, 3, 5의 곱
   - Bill (14) : 2, 7의 곱
   - Diagnosis (30) : 2, 3, 5의 곱
   - Dname (6) : 2, 3의 곱
   - Pname (42) : 2, 3, 7의 곱

아래 그림 5는 의료정보 XML 문서에 레이블링 기법을 적용한 결과이다. 루트 노드는 널 값이 부여되고, 첫 번째 자식 노드의 레이블은 "a1", 두 번째 자식노드는 "b1"이 부여된다. 부모 노드의 레이블이 자식 노드의 레이블에 상속된다. 두 노드 "2a1.a1.L3"과 "3a1a1.a1.L3"에 대해, "2a1.a1.L3"가 "3a1a1.a1.L3"의 접두사(prefix)이므로 "2a1.a1.L3"는 "3a1a1.a1.L3"의 부모 노드임을 알 수 있다.

3.2 '롤-기반 소수'에 의한 질의 처리

그림 6은 제안하는 접근제어 시스템의 구조이다. 그림 6에서 질의 처리 절차는 2 단계를 고려한다. 1단계에서 '롤-기반 소수 테이블'을 이용한 롤 검사와 2단계에서 '롤 프라이버시 테이블'을 이용한 최종 접근권한 검사이다. 일단 이동단말기에서 질의가 등록되면 1단계에서 질의를 등록한 이동 단말기의 롤-기반 소수를 검사한다. 즉, 1단계는 롤-기반 소수로 접근희망 노드의 접근 가능한 롤들의 곱(L3)을 나누어 그 나머지가 "0"이면 접근

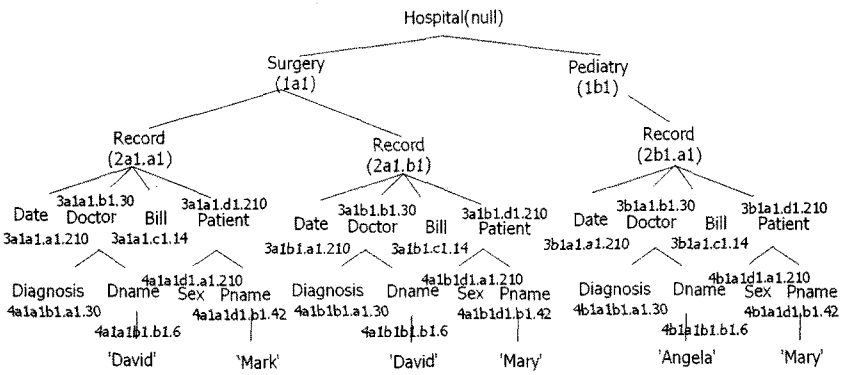

그림 5 제안하는 기법의 XML 레이블링 결과

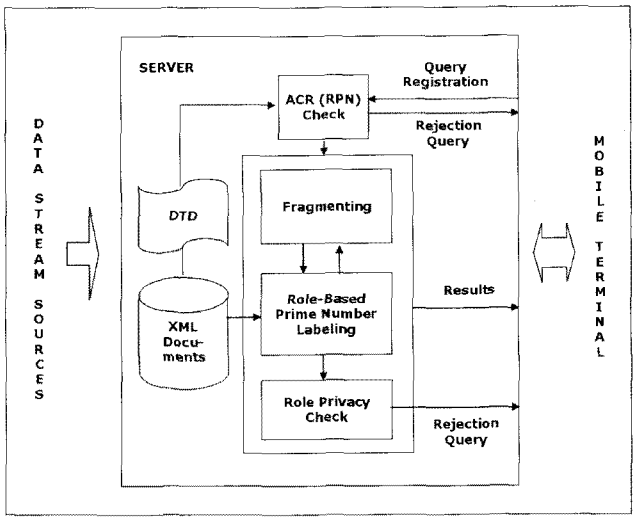

그림 6 제안하는 접근제어 시스템 구조

표 2 롤 프라이버시 테이블

Department	Record	Roles			
		Patient	Doctor	Insurer	Researcher
Sugery (a1)	a1.a1	Mark	David	ING	-
	a1.b1	Mary	David	AIG	-
	...	...	...	...	-
	...	...	...	...	-
Pediatry (b1)	b1.a1	Mary	Angela	AIG	-
	...	...	...	...	-
	...	...	...	...	-
...	...	...	...	...	-

을 허용하고 그렇지 않으면 질의를 거절한다. 그리고 2단계에서 롤 프라이버시 테이블을 참조하여 최종 접근 권한 여부를 검사한다. 또한, 접근 권한의 충돌 발생시 2장에서 언급한 '거절 우선정책'을 적용한다[6].

예제 1. (predicate + positive access control + positive access control)

(1) //record/patient[pname=Mark]

(2) doctor의 롤을 가진 David 이 질의 요청

-1단계, [pname=Mark] 노드 레이블 검색: '4a1a1d1.b1.42'

-David의 롤이 doctor(3) 이므로 42%3=0, 접근허용

-2단계, '롤 프라이버시 테이블'에 의하면 David은 'a1.a1'과 'a1.b1'으로 시작하는 노드 접근 허용

-[pname=Mark] 노드 접근 허용

-1단계 접근 허용 + 2단계 접근 허용; 질의 최종 허용

예제 2. (predicate + positive access control + negative access control)

(1) //record/patient[pname=Mark]

(2) doctor의 롤을 가진 Angela가 질의 요청

-1단계, [pname=Mark] 노드 레이블 검색: '4a1a1d1.b1.42'

-Angela의 롤이 doctor(3) 이므로 42%3=0, 접근허용

-2단계, '롤 프라이버시 테이블'에 의하면 Angela는 'b1.a1'으로 시작하는 노드에만 접근 허용

-[pname=Mark] 노드 접근 거절

-1단계 접근 허용 + 2단계 접근 거절; 질의 최종 거절

예제 3. (negative access control)

(1) //record/patient/pname

(2) researcher의 롤을 가진 누군가가 질의 요청

-1단계, pname 노드 레이블 검색: '*.*.42'

-researcher의 롤(5), 42%5≠0, 질의 거절

예제 3에서 보여준 것처럼 제안하는 방법의 주요 장점은 거절 질의에 대해 빠르게 응답한다.

3.3 갱신 레이블링 기법

이번 절에서는 트리 구조의 유지를 위해 자주 사용되는 엘리먼트, 텍스트 또는 속성의 삽입(insert)과 삭제(delete) 즉, 갱신(update)에 대하여 설명한다. 그러나 삭제 명령은 노드의 순서에 영향을 주지 않으며, 매우 단순한 처리를 하기 때문에 설명에서 제외하고 삽입 위주의 갱신에 대해 설명한다.

정의 1. (Insert Operation) 임의의 인접한 두 노드 N_{left}와 N_{right}에 대해 새로운 노드 N_{new}가 두 노드 사이에 삽입될 때 기존 노드의 레이블 수정없이 아래의 삽입 명령에 따라 실행된다. 여기서 len(N)은 노드의 bit 길이를 의미한다.

1. 첫 번째 형제 노드 앞에 삽입, N_{new} = 첫 번째 형제 노드(N_{left})의 마지막 비트 "1"을 "01"로 변경
2. 마지막 형제 노드 뒤에 삽입, N_{new} = 마지막 형제 노드(N_{right})의 마지막 비트 "1" + 1 (+는 합을 의미, 마지막 비트가 "9"이면 "9→91", 마지막 비트가 99이면 "99→991")
3. $len(N_{left}) \geq len(N_{right})$이면 $N_{new} = N_{left} \oplus 1$ ($\oplus$는 병합을 의미)
4. $len(N_{left}) < len(N_{right})$ 이면 N_{new} = N_{right}의 마지막 비트 "1"을 "01"로 변경

주어진 두 노드의 레이블 "a01"과 "a1"에 대해서 "a01"<"a1"과 같은 사전적 순서(lexicographically order)가 정해지는데 그 이유는 왼쪽 비트부터 차례대로 비교할 때, "a01"의 두 번째 비트 "0"가 "a1"의 두 번째 비트 "1" 보다 선행하기 때문이다. 다른 예로, "a1"와 "a10"을 보면 "a1"<"a10"이다. 왜냐하면 "a1"이 "a10"의 접두사이기 때문이다.

정의 2. (Lexicographical order <) 임의의 두 노드 레이블 N_{left}와 N_{right}가 아래와 같은 조건에 대해 N_{left}는 N_{right} 보다 사전적 순서($N_{left} < N_{right}$)가 선행한다고 한다.

1. N_{left}는 N_{right}의 접두사이거나,
2. N_{left}와 N_{right}의 비트 스트링을 비교해서 만약 N_{left}의 현재 스트링 Ncurrent와 N_{right}의 현재 스트링 Ncurrent가 "<"을 만족하면, $N_{left} < N_{right}$이고 비교는 종료된다[6].

두 노드 "1a1"과 "1b1" 사이에 새로운 노드를 삽입하고자 할 때, "1a1"과 "1b1"의 길이는 동일하다. 따라서 "1a1" 뒤에 1을 병합($\oplus$)한다 (알고리즘1의 라인 1과 2 참조). 새롭게 삽입된 노드 레이블 "1a11"은 "1a1" < "1a11" < "1b1"으로 사전적 순서를 갖는다. 두 노드 "1a11"과 "1b1" 사이에 새로운 노드를 삽입 하고자 할 때 "1a1"의 길이는 3이고 길이가 4인 "1a11" 보다 작다. 그러므로 "1a11" 노드의 마지막 비트 "1"을 "01"로 변경한다. 즉, "1a101"이 된다 (알고리즘1의 라인 3과 4 참조). 여기서도 "1a1" < "1a101" < "1a11" 사전적 순서

가 지켜짐을 알 수 있다. 두 노드 "1a11"과 "1b1" 사이에 새로운 노드를 삽입 하고자 할 때, "1a11"의 길이는 4이고 "1b1" 보다 길다. 따라서 "1a11" 뒤에 "1"이 병합되어 "1a111"이 된다(알고리즘1의 라인 1과 2 참조). 삽입된 레이블 "1a111"은 "1a11" < "1a111" < "1b1" 사전적 순서를 유지한다.

본 논문에서 제안한 동적 롤-기반 소수 레이블링 기법은 그림 7의 예에서처럼 XML 문서가 갱신될 때 갱신비용을 발생시키지 않는다.

알고리즘 1: Insert Between (N_{left}, N_{right})

Input: $N_{left} < N_{right}$
Output: N_{new} such that $N_{left} < N_{new} < N_{right}$ lexicographically
Description:
1 : if len(N_{left}) ≥ len(N_{right}) then
2 : $N_{new} = N_{left} \oplus 1$ // ⊕ means concatenation
3 : else if len(N_{left}) < len(N_{right}) then
4 : N_{new} = N_{right} with the last bit " 1" changed to " 01"
5 : end if
6 : return N_{new}

3.4 노드관계

자식 노드의 레이블을 생성할 때 부모 노드의 레이블을 사용함으로써 노드들 사이에 조상-후손 관계와 형제 관계를 결정하는데 유용하다. 예를 들어, 그림 7에서처럼 "2a1.a1"의 부모 노드는 "1a1"이고 레벨 2로 시작하는 노드는 형제 노드이며, "2a1.a1"의 모든 자식 노드는 "3a1a1"으로 시작함을 알 수 있다.

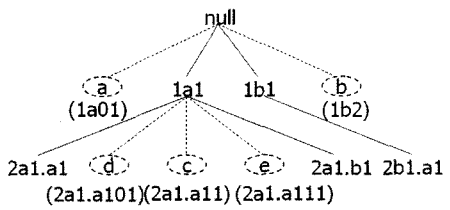

그림 7 동적 갱신 레이블링의 예

제안하는 레이블링 기법의 또 하나의 장점은 트리 구조의 깊이(depth)를 레이블의 레벨을 통해 쉽게 알 수 있다. 이것은 각 노드 레이블의 첫 번째가 레벨을 의미하기 때문이다. 레이블링 기법에서 특정 레벨을 빠르게 찾아가는 레벨 넘버를 사용함으로써 검색, 삽입, 삭제 또는 갱신 명령을 더욱 수월하게 실행하도록 해준다.

4. 실험 및 성능 평가

이 장에서 다양한 방법으로 제안하는 레이블링 기법을 검증하고자 하였다. 우선, 주기억 장치 1MB를 가진 Intel Pentium IV 2.66GHz CPU의 하드웨어와 운영체제는 MS Windows XP Professional을 사용하였으며, 제안하는 레이블링 기법의 구현은 JAVA (JDK1.5.0)와 XML 파싱을 위해 SAX Parser를 사용하였으며 데이타 생성은 XMark를 사용하였다[21].

4.1 레이블

동적 XML 데이타를 지원하는 GRP(GRoup based Prefix labeling)의 성능을 분석하고, [19]에서 사용했던 방식으로 XMark를 이용하여 10개의 XML 문서를 생성하였다.

제안하는 방법으로 레이블링을 수행하여 다른 레이블링 방법(GRP)과 레이블의 총 길이를 비교하였다. 그림 8에서 처럼 제안 하는 레이블링 기법이 GRP 보다 평균 3배의 레이블 사이즈를 절약함을 알 수 있었다.

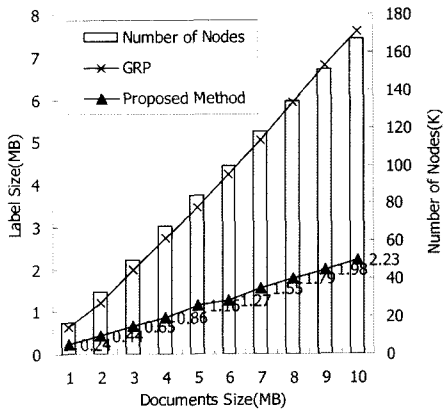

그림 8 레이블의 총길이

그리고, 그림 9에서처럼 레이블링 초기 생성 시간에서도 기존 방법(GRP) 보다 성능 면에서 우수함을 알 수 있었다. 또한, XML 데이타에 대한 통상적인 질의 처리를 위한 문서 검색 시간 단위를 100으로 가정할 경우 3.1절에서 언급한 XML 문서의 단편화를 이용할 경우 3.2절의 예제 1의 예에서 30%, 예제 2의 예에서 60%, 예제 3의 예에서 100%의 문서 검색을 줄일 수 있었다.

레이블의 갱신 측면에서도 그림 10에서 보이듯이 소수 레이블링 기법[20]은 노드 삽입시 레이블의 갱신을 위한 노드의 참조 또는 갱신이 상당 수 발생됨을 알 수 있다. 그러나 제안하는 레이블링 기법은 갱신을 위해 다른 노드의 참조 또는 갱신이 전혀 발생되지 않는다.

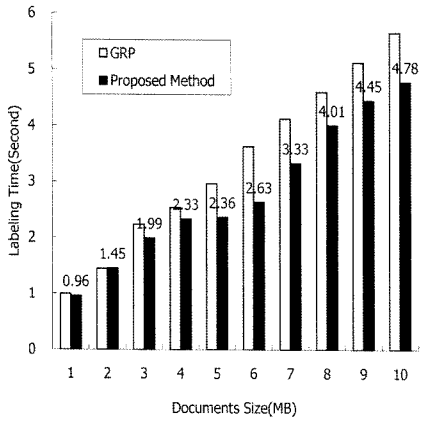

그림 9 레이블의 초기 생성시간

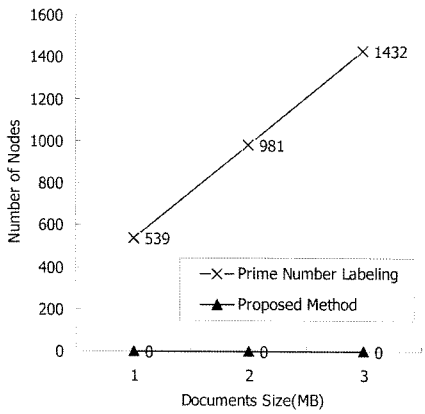

그림 10 갱신시 다시 레이블링되는 노드의 수

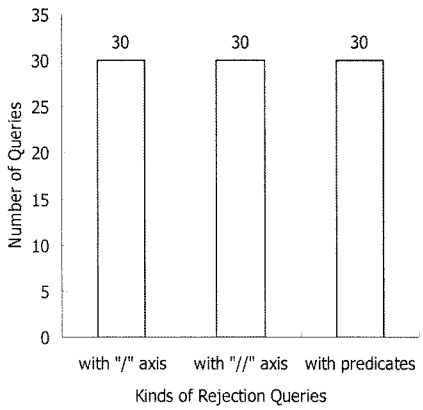

그림 11 거절 질의의 검출

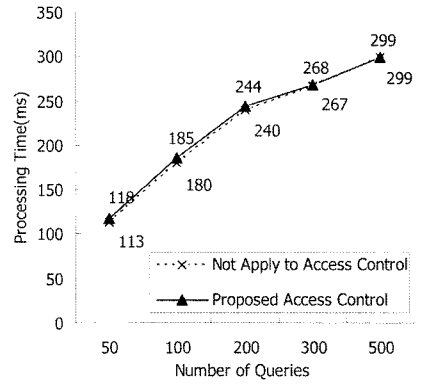

그림 12 안전한 질의의 처리시간

4.2 질의처리

거절질의는 어떤 경우라도 거절 되어야 하는 접근제어 정책에 부적합한 질의를 의미한다. 실험에서 30개의 의도된 거절질의를 작성하여 접근제어 정책을 적용하였을 때 정확하게 거절질의를 검출하는지를 확인하였다.

그림 11은 거절질의에 대한 실험 결과를 말해준다. 실험은 3가지("/" 축, "//" 축, 그리고 프레디키트를 가진 단말노드) 경우로 실행되었다. 그 결과 30개의 거절 질의에 대해 100% 모두 검출해내는 성능을 보여주었다.

평균 질의처리 시간은 2가지 경우를 비교하였다. 즉, 접근제어 정책을 적용했을 때 그리고 적용하지 않았을 때를 비교하였다. 평균 질의처리 시간은 임의의 XPath 질의의 수를 각각 50, 100, 200, 300, 그리고 500개로 하여 실행하였다. 평균 질의처리 시간은 오차를 최소화하기 위해 10회의 측정을 통해 최대, 최소값을 생략하고 나머지 값들의 평균시간으로 측정하였다. 접근제어 설정 시간은 질의 처리 시간에 포함시키지 않았다. 왜냐하면, 문서의 삽입과 삭제 같은 갱신은 질의 처리시 발생하는 시간으로 보기 보다는 선처리 시간으로 판단하여 제외하였다. 동적 롤-기반 소수 레이블링 시간도 질의 처리 시간 전에 생성되기 때문에 제외시켰다. 그럼에도 불구하고 접근제어를 판단하는 시간은 그림 12에서처럼 시스템의 질의처리 성능에 큰 영향을 미치지 못했다.

5. 결 론

본 논문은 동적 XML 데이타 스트림 환경에서 효율적이고 안전한 접근제어 레이블링을 제안하였다. 또한, 기존 연구에서 빈번히 갱신되는XML문서의 접근제어와 레이블링 기법의 문제점을 지적하였다. 동적 롤-기반 소수 레이블링 기법은 노드들 사이의 조상-후손 관계와 형제 관계를 쉽게 알 수 있으며, 기존 노드의 레이블 갱신이 발생하지 않는다. 또한, 제안하는 레이블링 기법은 임의의 노드 사이에 충돌없이 삽입이 가능하며, 무한 갱신 환경 속에서도 지원이 가능하다. 본 논문은 의료정보 XML 문서와 같이 제안하는 환경에서 병원 환자의 증가로 문서의 깊이보다 폭이 무한히 증가될 때 더욱 유용하며, 이러한 방법을 제안하는 방법을 적용할 경우 충분히 활용 할 수

있다. 보안 측면에서도 2단계 보안을 적용하여 빠른 거절질의 탐색을 통한 시스템의 부하 최소화와 세밀한 접근제어 정책을 적용하여 보안을 더욱 강화 하였다.

향후에는 본 연구에서 제안하는 환경인 의료정보 XML 문서를 일반화하여 일반 XML 문서에서도 효율적으로 적용하고자 한다.

참 고 문 헌

[1] S. Berglund, D. Boag, M.F. Chamberlin, et al, "XML path language (XPath) 2.0," W3C working draft 16. Technical Report WD-xpath20-20020816, World Wide Web Consortium, 2002.

[2] S. Boag, D. Chamberlin, M. F. Fernandez, D. Florescu, J. Robie, and J. Simon, "XQuery 1.0: An XML Query Language," W3C Working Draft 16. Technical Report WD-xquery-20020816, World Wide Web Consortium, 2002.

[3] B. Babcock, S. Babu, M. Datar, R. Motwani, and J. Widom, "Models and Issues in Data Stream Systems," In PODS(Symposium on Principles of Database Systems), 2002.

[4] B. Carminati, E. Ferrari, K. L. Tan, Specifying Access Control Policies on Data Streams Outsourced Data, In DASFAA(Database Systems for Advanced Applications), 2006.

[5] C. Li, T.W. Ling, "QED: A Novel Quaternary Encoding to Completely Avoid Re-labeling in XML Updates," In CIKM(ACM Conference on Information and Knowledge Management), 2005.

[6] C. Li, T. W. Ling, M. Hu, "Efficient Processing of Updates in Dynamic XML Data," In ICDE(IEEE International Conference on Data Engineering), 2006.

[7] P.E. O'Neil, E.J. O'Neil, S. Pal, I. Cseri, et al. "ORDPATHs: Insert-Friendly XML Node Labels," In SIGMOD(ACM International Conference on Management of Data), 2004.

[8] J. Xu, Z. Li, Y. Wang, R. Yao, "An Efficient Encoding and Labeling Scheme for Dynamic XML Data," In DEXA(International Conference on Database and Expert Systems Applications), 2007.

[9] E. Bertino, S. Castano, E. Ferrari, and M. Mesiti, "Specifying and Enforcing Access Control Policies for XML Document Sources," In WWW Journal, 2000.

[10] E. Bertino and E. Ferrari, Secure and Selective "Dissemination of XML Documents," In TISSEC (ACM Transactions on Information and System Security), Vol.5, No.3, 2002.

[11] E. Damiani, S.Vimercati, et al. "Securing XML Document," In EDBT(International Conference on Extending Database Technology), 2000.

[12] E. Damiani, S. Vimercati, et al., "Access Control System for XML Documents," In TISSEC(ACM Transactions on Information and System Security), Vol.5, No.2, 2002.

[13] M. Murata, A. Tozawa, and M. Kudo, "XML Access Control Using Static Analysis," In CCS (ACM Conference on Computer and Communications Security), 2003.

[14] B. Luo, D. W. Lee, W. C. Lee, and P. Liu, "Qfilter: Fine-grained Run-Time XML Access Control via NFA-based Query Rewriting," In CIKM(ACM Conference on Information and Knowledge Management), 2004.

[15] http://www.w3.org/TR/xml-fragment

[16] S. Bose, L. Fegaras, "XFrag: A Query Processing Framework for Fragmented XML Data," In Web and Databases, 2005.

[17] H. Huo, G. Wang, X. Hui, R. Z. BoNing, and C. Xiao, "Efficient Query Processing for Streamed XML Fragments," In DASFAA(Database Systems for Advanced Applications), 2006.

[18] W. Fan, I. Fundulaki, F. Geerts, X. Jia, A. Kementsietsidis, "A View Based Security Framework for XML," In AHM(All Hands Meeting Home), 2006.

[19] J. Lu, T. Wang Ling, "Labeling and Querying Dynamic XML Trees," In APWeb(Asia-Pacific Web Conference), 2004.

[20] X. Wu, M. Li, L. Hsu, "A Prime Number Labeling Scheme for Dynamic Ordered XML Trees," In ICDE(IEEE International Conference on Data Engineering), 2004.

[21] A. Schmidt, F. Waas, M. Kersten, M. J. Carey , I. Manolescu, R. Busse, "XMark: A Benchmark for XML Data Management," In VLDB(Very Large Data Base), 2002.

안 동 찬

2004년 서강대학교 컴퓨터공학과 공학박사 수료. 2002년~현재 안산공과대학 디지털미디어과 교수. 관심분야는 접근제어, 이동 데이타베이스, 웹 데이타베이스, 컴퓨터 보안

박 석

정보과학회논문지 : 데이타베이스 제 36 권 제 2 호 참조