산업기술연구(강원대학교 산업기술연구소 논문집), 제23권 A호, 2003
Journal of Industrial Technology, Kangwon Natl. Univ., Korea, No 23 A, 2003.

XML 문서의 효율적인 미세 접근 제어 시스템을 위한 명시적 접근 권한 트리 생성 기법

An Explicit Access Authorization Tree Generation Technique for the Efficient Fine-Grained Access Control System of XML Documents

이 헌 길* 강 정 모**
Lee, Heon-Guil Kang, Jung-Mo

Abstract

차세대 웹 문서의 새로운 표준으로 자리 잡아 가고 있는 XML은 전자 상거래나 병원 관리 등과 같이 다양한 응용 분야에 적용되고 있다. 이러한 응용 분야는 XML 문서의 각 구성 요소 수준의 세밀한 액세스 제어를 요구한다. 따라서, 현재 XML 문서의 미세 접근 제어 기법에 대한 연구가 활발히 이루어지고 있다 현재 제안된 XML 문서를 위한 미서 접근 제어 기법은 대부분 묵시적 권한 기법을 사용하여 상위 구성요소에 한번의 권한 부여로 하위 노드에 권한을 부여하는 효과를 가지는 장점을 가지나, 각 구성 요소간의 권한을 체크하는 시간 오버헤드를 가지는 단점이 있다 본 논문에서는 권한을 체크하는 시간을 감소시키기 위하여 XML 문서의 미세 접근 제어를 위한 명시적 권한 기법을 제안하고, 이를 이용하여 노드의 권한에 대한 모든 정보를 저장하고 있는 접근 권한 트리 모델을 제시하였다 본 논문에서 제안한 접근 권한 트리 모델을 사용하면, 권한 정보를 저장하는 공간 오버헤드가 있지만, 권한 체크 시간을 감소시켜 사용자에게 보다 빠른 뷰를 제공할 수 있다

키워드 : *XML, 명시적 권한 기법, 미세 접근 제어*

1. 서론

차세대 웹 문서의 새로운 표준으로 자리 잡아 가고 있는 XML은 HTML과 달리 사용자 정의 태그가 가능하기 때문에 문서의 논리적 구조를 표현할 수 있고, SGML의 복잡성을 단순화시켜 웹상에서 다양한 문서 구조를 표현할 수 있는 장점을 가지고 있다[15][16][19] 이에 따라 XML은 현재 병원관리나 전자 상거래 등의 다양한 응용분야에 적용되고 있다 이와 같은 응용 분야에서 XML 문서의 구성요소는 이름 등과 같이 공개되어도 상관없는 일반적인 정보와 주민등록번호, 전화번호 등과 같이 타인에게 공개되지 말아야 하는 개인 정보로 나눌 수 있다 이로 인해 XML 문서 내 서로 다른 권한을 가지는 각 구성요소에 대한 세밀한 액세스 제어를 하기 위하여 미세 접근 제어(fine-grained access control) 기법이 사용된다[8][14] 이에 따라 최근 XML에 각종 권한 기법을 적용하여, 미세 접근 제어를 제공하는 기법들이 활발히 제안되고 있다[2][6][7]

현재 XML의 접근 제어 기법 중 많은 연구가 이루어지고 있는 묵시적 권한 기법은 상위 구성요소에 한 번 권한을 부여함으로써 하위 구성요소에 권한을 부여하는 효과를 가지는 장점을 가지고 있다. 하지만, 권한을 가진 구성요소에 접근할 경우와 권한을 추가로 부여하는 경우 상위 구성요소와의 충돌 유무를 판별하는 권한 체크시간이 많이 걸리는 단점이 있다[23][24]. 반면, 명시적 권한 기법은 각 구성요소 마다 권한 정보를 저장하는 기법이다 명시적 권한 기법은 묵시적 권한 기법에 비하여 저장 공간의 오버헤드가 있지만, 각 구성요소가 권한 정보를 가지고 있으므로 즉각적인 접

* 강원대학교 컴퓨터정보통신공학과 교수, 공학박사
** 강원대학교 컴퓨터정보통신공학과 석사과정

근 제어를 할 수 있다. 그리고 묵시적 권한 기법보다 권한 체크 시간이 적게 걸리므로, 권한 체크가 자주 발생하는 응용분야에 적용하면 권한 체크 시간 오버헤드를 감소시킬 수 있다.

본 논문은 권한을 체크하는 시간을 감소시키기 위하여 XML 문서를 위한 명시적 권한 기법을 제안하고, 이를 이용하여 노드의 권한 정보를 가지고 있는 접근 권한 트리 모델을 제시하였다. 전자 상거래나 병원 관리 등의 응용분야에서는 문서에 대한 권한 정책이 자주 수정되지 않고, 읽기에 대한 비율이 쓰기에 대한 비율보다 현저하게 많다. 이러한 응용 분야에 본 논문에서 제시한 트리 모델을 이용하면, 권한 체크 시간을 감소시켜 사용자에게 보다 빠른 뷰(view)를 제공할 수 있다.

본 논문의 2장은 연구 배경 및 관련 연구를 기술한다. 3장에서는 XML 문서의 효율적인 미세 접근 제어 시스템을 위한 명시적 권한 기법, 문서의 구성 요소가 가질 수 있는 권한 정보를 저장하는 접근 권한 트리 모델, 그리고 권한 트리 생성 알고리즘을 제시한다. 4장에서는 제안된 기법의 성능을 분석하였다. 마지막으로 5장에서 결론을 맺는다.

2. 연구배경 및 관련연구

2.1 미세접근 제어를 위한 권한기법

접근 제어 기법은 거시 접근 제어(coarse-grained access control)기법과 미세 접근 제어 기법의 두 가지로 나눌 수 있다[4][21]. 거시 접근 제어 기법은 전체 XML 문서 단위로 접근 제어가 이루어지므로, 세밀한 접근 제어가 어렵다. 반면 미세 접근 제어 기법은 XML 문서 내의 documentElement, 노드 등과 같은 구성 요소 단위로 접근 제어를 하는 방식으로써 계층적인 항목에 대하여 세밀한 접근 제어를 할 수 있다[5]. 미세 접근 제어를 위한 권한 기법은 명시적 권한 기법과 묵시적 권한 기법이 있다[10]21][23][24][25].

2.2 XML 문서를 위한 미세 접근 제어 기법

2.2.1 Tokyo IBM Research Lab(TIRL)

Tokyo IBM Research Lab에서는 XML 문서를 위한 미세 접근 제어 기법을 최초로 제안하였다[1]. 이 모델에서는 미세 접근 제어를 실시하기 위하여 권한 기법 중 묵시적 권한 기법을 사용하였고, 공개키 암호화 방식을 사용하여 문서를 암호화시켜 전송하는 기법을 제안하였다. 제안된 모델은 권한을 subject, object, action, condition, policy, policy evaluation으로 표기하고, 미세 접근 제어를 하기 위하여 권한의 전송 정책이나 충돌 탐지 정책 등과 같은 다양한 정책들을 제안하였다. 이 모델에서 제안한 표기법과 각종 정책들은 XML 문서를 위한 다른 여러 미세 접근 제어 기법에 활용되고 있다[2][6]. 하지만, 이 모델은 시스템 구현적인 측면보다 XACL(XML access control language)을 이용한 접근 제어 표현 방식에 중점을 두었다.

2.2.2 Author-X

Author-X[3]는 그림 1과 같이 사용자의 권한 등급에 따른 암호화된 키를 분배하여 미세 접근 제어를 시도한 시스템으로서, Bertino에 의하여 만들어 졌으며, Tokyo IBM Research Lab의 방식 중 공개키 암호화 방식에 중점을 두어 설계하였다.

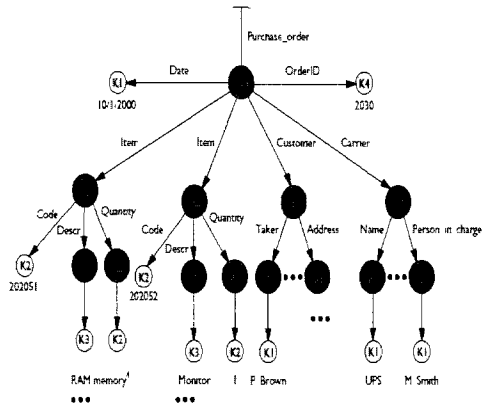

그림 1 Author-X의 키 생성

Author-X 시스템에서는 크게 두 가지 모드 방식으로 권한 접근 제어를 관리한다. 접근 권한의 유무를 판별하는 Pull 모드 방식과 변경된 내용에 대하여 다른 키를 사용자에게 부여하는 Push 모드 방식이 그것이다. 하지만 Author-X는 너무 많은 키를 관리해야 하는 단점을 가진다

2.3.3 Damiani 모델

Damiani 모델[5][8][9][13]은 Tokyo IBM Research Lab의 방식 중 묵시적 권한 기법에 중점을 두어 설계하였으며, XML의 구조와 묵시적 권한 기법의 특성에 따라 표 1과 같이 권한을 8가지의 타입으로 분류하였다

표 1 권한 타입

Level /Strength	Propagation	
	Local	Recursive
Instance	L	R
Instance(Soft statement)	LS	RS
Schema	LD	RD
Schema (Hard statement)	LDH	RDH

권한의 레벨에 따라 Schema/Instance로, 권한이 명시적인지 묵시적인지에 따라 Local/Recursive로, 권한의 강도에 따라 Hard/Soft로 분류하였다[20] 제안된 Damiani 모델은 XML 문서의 응용 분야 중 읽기에 대한 비율이 쓰기에 대한 비율보다 현저하게 많은 분야를 적용 대상으로 하였다[5]. 그리고 사용자가 문서를 요청하면 권한을 체크하여, 권한을 가진 문서의 일부분만을 제공하는 뷰 생성 알고리즘을 제시하였다[9][12]

이게 Damiani 모델의 뷰 생성 알고리즘을 분석하기도 한다. 먼저, n을 XML 문서 내의 전체 구성 노드의 수라고 하면 Damiani 모델의 뷰 생성 알고리즘에 대한 시간 복잡도는 다음과 같다 사용자가 뷰를 요구할 때 마다 매번 뷰 생성 알고리즘의 set label과 getfinal label 프로시저에서 두 번의 전체적인 트리 탐색($2n$)과 최종적으로 권한이 부여되어진 트리만을 찾아가는 과정(n)이 이루어져야 하므로, 뷰를 만드는데 걸리는 시간은 최선의 경우 $2n$이고, 최악의 경우 $3n$이다 최선의 경우는 권한을 가지는 최상위 노드(루트 노드)가 부정으로 지정되어 이후의 트리 탐색이 이루어지지 못하는 경우이며, 최악의 경우는 모든 노드의 권한이 긍정으로 지정되어 뷰를 생성하는데 모든 노드를 탐색해야 하는 경우이다. 결과적으로 Damiani 모델은 평균적으로 $\frac{5n}{2}$ 만큼의 시간 복잡도를 가진다. 또한, 사용자가 XML 문서를 요청할 때 마다 매번 뷰 생성 알고리즘을 수행하여 뷰를 계산해야 한다.

Damiani 모델에 대한 공간 복잡도는 다음과 같다. 먼저, n을 권한 정보를 표현하는데 필요한 XML DTD의 노드 수라고 가정한다 Damiani 모델에서는 묵시적 권한 정보를 표현하는데 필요한 저장 공간만 있으면 된다 이유는 한번의 권한 부여로 하위의 구성요소에 권한을 부여하는 효과를 갖는 묵시적 권한 기법을 사용하기 때문이다 그러므로 공간 복잡도는 $O(1)$이 된다

3. 제안된 권한 기법

3.1 XML 문서를 위한 명시적 권한 기법

XML 문서를 위한 명시적 권한 기법은 XML 문서를 위한 효율적인 미세 접근 제어와 Damiani 모델의 단점인 권한 계산의 오버헤드를 감소시켜 사용자에게 빠른 뷰를 제공하기 위하여 제안되었다 또한, 본 논문은 문서를 관리하는 권한 정책과 문서 내 각 노드가 가지고 있는 모든 정보를 계산하여 명시적 접근 권한 트리에 저장하는 방식을 제시하고 있다. 접근 권한 트리는 사용자에게 뷰를 제공할 때 사용되는데, 전자 상거래나 병원 관리와 같이 문서에 대한 권한 정책이 자주 수정되지 않고, 읽기에 대한 비율이 쓰기에 대한 비율보다 현저하게 많은 응용분야에 적용하면 사용자에게 보다 빠른 뷰를 제공할 수 있다.

XML 문서를 사용하는 각 응용 분야에서는 그 분야에 대한 권한 정책들을 가지고 있다. 다음의 표 2에서는 본 논문에서 예로 제시할 병원 관리에 대한 권한 정책 예를 보여 준다 이 권한 정책들은 표 3에 제시한 권한 정책 테이블에 저장되어 관리된다[8]. 각 권한 정책의 후미에 나오는 [a] ~ [r]은 표 3에 나오는 권한 정책 테이블에 표기되는 권한명을 의미한다 뷰를 만들 때 마다 매번 권한 정책을 파싱하여 사용하는 Damiani 모델과 달리 제안된 권한 기법은 권한 정보를 미리 한번 파싱하여 권한 정책 테이블에 저장하고 뷰를 계산할 때 사용한다

권한 정책 테이블은 subject, object, action, sign으로 구성되어 있다. subject는 user-group, ip-address, symbolic name의 3가지 구성요소로 이루어져 있다[8] user-group은 각 적용분야에 접근하는 그룹의 다양한 종류로 표현된다. ip-address는 210.115.48.*등과 같은 ip 주소이며, symbolic name은 hospital.com과 같은 인터넷 주소를 의미한다 subject(public, 210.115.48.*, hospital.com)는 210.115.48.*의 ip 주소로 hospital.com을 통하여 접속하는 모든 사람을 의미한다. object는 권한이 표현되는 각 구성요소를 나타내며, XML 문서 내에서는 각 노드에 해당한다 object의 표현은 XPATH[17]를 이용한다. 의미를 간단히 살펴보면, 아래의 표 3에서 [a]라는 권한의 object는 병원관리 XML 문서의 department 내의 name이라는 노드를 가리킨다 action은 읽기라는 하나의 행위이다 본 모델은 전자 상거래나 병원 관리 등과 같이 읽기에 대한 비율이 쓰기에 대한 비율보다 현저하게 많은 응용분야를 가정하여 만들어졌으므로, action은 읽기만으로 정의한다 sign은 +, -로 표현된다.

표 2 병원관리 응용을 위한 권한 정책 예

1 department 내 name은 모든 사람이 접근할 수 있다(public accessed) [a]-[b]

2. 의료진과 환자의 주소에 대한 정보는 도메인 *hospital.com으로 접속하는 관리자 그룹의 멤버만이 접근할 수 있다. [c]-[d]

3 의료진의 봉급은 host 159.101.80.5로부터 접근하는 관리자 그룹의 멤버만이 접근할 수 있고[e], 모든 환자의 치료비는 *hospital.com으로 접속하는 관리자 그룹의 멤버만이 접근할 수 있다.[f] 그리고, 이를 제외한 모든 사람들은 절대로 의료진의 봉급과 환자 치료비 정보를 접근할 수 없다 [g]-[h]

4 의료진에 대한 정보는 (그들의 봉급과 집 주소를 제외하고) 모든 사람에게 접근이 허용된다. [i]-[j]-[k]

5 환자에 대한 정보는 의료진에게만 접근이 허용된다. [m]-[n]

6 의약 연구에 대한 정보는 병원의 의료진에게만 접근이 허용된다 [o]-[p]

7 연구 과제 중 project1은 host 159로 시작하는 의사에게만 접근이 허용된다 [q]

8 환자의 이름에 대한 정보는 도메인이 *hospital.com으로 접속하는 관리자 그룹의 멤버만이 접근할 수 있다. [r]

표 3 권한 정책 Table 예

정의	Subject	Object(Path expression)	Action	Sign
a	Public.*.*	/department/@name	read	+
b	Public.*.*	/department/division	read	+
c	Public.*.*	/department//name	read	+
d	Administrative.*.*.hospital.com	/department//address	read	+
e	Administrative.159.101.80.5.*	/department//salary	read	+
f	Administrative.*.*.hospital.com	/department/patient//cost	read	+
g	Public.*.*	/department/medicalStaff//salary	read	-
h	Public.*.*	/department/patient//cost	read	-
i	Public.*.*	/department/medicalStaff	read	+
j	Public.*.*	/department/medicalStaff//address	read	-
k	Public.*.*	/department/medicalStaff//salary	read	-
m	MedicalStaff.*.*	/department/patient	read	+
n	Public.*.*	/department/patient	read	-
o	MedicalStaff.*.*	/department/research	read	+
p	Public.*.*	/department/research	read	-
q	Physician.159.*.*	/department/research/project1	read	+
r	Administrative.*.*.hospital.com	/department/patient//name	read	+

3.2 명시적 접근 권한 트리

명시적 접근 권한 트리는 사용자가 요청하는 문서와 사용자의 권한, 그리고 권한의 정보를 가지고 있는 권한 정책 테이블을 이용하여, 문서 내의 각 노드가 가지고 있는 모든 권한 정보를 트리 형태로 저장하고 있는 것을 말한다. 이후, 사용자가 뷰를 요청할 때 이 접근 권한 트리를 이용하여 효율적으로 사용자에게 뷰를 제공하게 된다.

표 4 병원 관리 XML 문서의 예

```
<?xml version="1.0"?>
<!DOCTYPE Hospital_Department SYSTEM"dept.dtd">
<department name="Medicine">
  <division> Cardiology </division>
  <medical_staff>
    <physician>
      <name> Bob </name>
      <specialty> Nuclear Cardiology </specialty>
      <office> CD393 </office>
      <phone> 415-5555 </phone>
      <address>
        <street> 25 cherry Ave </street>
        <city> Emeryville </city>
      </address>
      <salary> $ 30,000 </salary>
    </physician>
    <nurse>
      <name> Tina </name>
      <address>
        <street> 14th St </street>
        <city> oaKland </city>
      <address>
      <salary> $ 20,000 </salary>
    </nurse>
  <medical_staff>
  <research> ... .... . . . </research>
  <patient> ..... .. .... . . </patient>
</department>
```

그림 2는 본 논문에서 예로 제시된 병원 관리를 위한 XML 문서인 표 4에 대한 명시적 접근 권한 트리로서 접근 권한 트리의 각 노드에는 그 노드가 가질 수 있는 권한의 모든 경우를 저장하고 있다. 접근 권한 트리 내의 권한은 권한의 주체인 sub ject와 권한의 sign의 두 가지의 구성 요소로 표현된다. subject는 user-group, ip-address, symbolic-address의 집합으로 이루어져 있다. sign은 +/-로 표현되며, 권한의 긍정과 부정을 나타낸다.

3.3 명시적 접근 권한 트리 생성 알고리즘

본 절에서 제안하는 명시적 접근 권한 트리 생성 알고리즘은 권한 정책과 XML 문서를 이용하여 XML 문서 내의 각 구성요소에 권한 정보를 표시한 권한 트리를 생성하는 알고리즘이다. 제안한 알

고리즘은 크게 세 단계로 이루어져 있다 첫 번째 단계는 권한 정책 테이블에 명시된 명시적 권한을 일단 트리 내에 저장하는 단계이다. 두 번째 단계는 명시되지 않은 구성요소에 대하여 명시적으로 권한을 저장시키는 단계이다. 그리고 마지막 단계는 권한 충돌을 방지하기 위한 단계이다. 권한이 없는 구성 요소에 대하여 권한을 부여하는 두 번째 단계의 자세한 과정은 다음과 같다 하위의 트리로 탐색하다가 하위 구성 요소에 상위에서 부여되는 권한과 같은 subject에 대하여 명시적 권한이 부여되어 있을 경우, 권한 부여를 중단하고, subject가 다르면 권한 부여를 계속 하게 된다 명시적 권한 기법은 묵시적 권한 기법과 달리 구성요소마다 권한을 명시적으로 저장하므로, 권한을 계산하는 단계에서 발생하는 권한 충돌이 거의 일어나지 않는다. 하지만 상위 구성 요소의 권한이 -이고, 하위 구성 요소의 권한이 +이면 트리에 모순이 발생하게 되어 권한 충돌이 일어나게 된다. 이를 방지하기 위하여 큐(queue)를 이용하여 권한 부여의 역순으로 즉, 하위 구성 요소부터 상위로 거슬러 올라가며 권한을 비교하여, 충돌이 있는지 여부를 판정한다. 그림 3은 명시적 접근권한 트리 생성 알고리즘과 자료구조이다

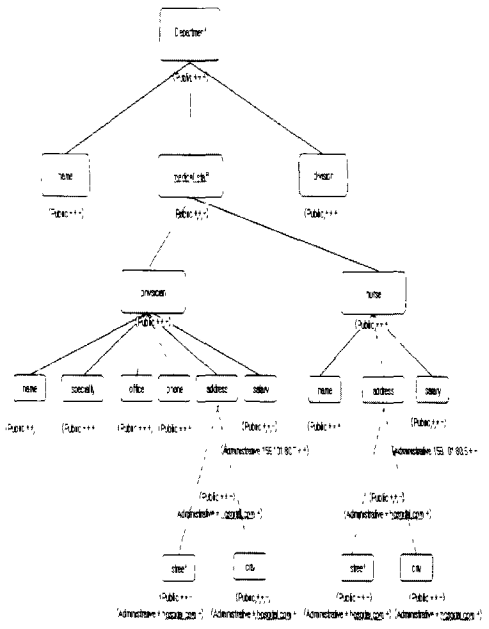

그림 2 의료진에 대한 명시적 접근 권한 트리 예

- P : 권한 정책(Policy)
- D : XML 문서
- N : 임의의 노드, n의 상위 노드
- n : N의 하위노드
- R : 루트 노드
- A : 노드 n의 접근 권한 (subject, sign)
- subject : 권한 주체
- sign : +(긍정), -(부정)
- Q : 노드들의 집합(queue)
- S : 노드들의 집합(stack)
- i, j : 임의의 정수 값

```
struct Policy {
    subject :
    object ,
    sign :
} P ,
```

```
struct Access_Authorization {
    subject ,
    sign ,
} A ,
```

```
Input  권한 정책 P[]와 XML 문서
Output  명시적 Access Authorization Tree
Algorithm 명시적 Access Authorization Tree 생성
begin
N = R .
// 정책 테이블에 명시된 정책을 문서내의 노드에 설정한다
1 권한 정책 P의 모든 P[i]에 대하여 P[i].subject와 P[i].sign을 D의
  P[i].object에 추가 .
2  N의 A가 설정되지 않았다면, (public,*,+)으로 설정 .
// 각 구성요소에 명시적 권한을 부여 한다
while (Q is not empty)
{
    3 N = Delete(Q) .
    4 N의 A[i]와 n의 A[k]에 대하여 다른 subject를 비교 .
      3.1 subject가 다르다면 N의 A[i]를 n의 A에 추가 .
        // subject에 대한 권한 정책이 이미 존재한다면
                그것을 유지한다
      3.2 subject가 동일하다면 기존의 권한을 유지 :
    5 Add(Q, n) :
    6 push(S, N) .
}
// 충돌 탐지
while (S is not empty)
{
    7 n = pop(S) .
    8 n의 A[i]와 N의 A[k]  subject가 동일하고, A[i]의 sign이 '-'
      이고, A[k]의 sign이 '-' 라면, A[k]의 sign을 '+'로 설정 .
}
end
```

그림 3 명시적 접근 권한 트리 생성 알고리즘

4. 제안된 기법 분석

본 논문에서 제안된 기법에 대하여 뷰를 생성하는 데 걸리는 시간과 저장 공간 복잡도에 대해 분석하기로 한다

먼저, n을 XML 문서 내의 권한 정보를 가질 수 있는 전체 노드의 수라고 가정하면 제안된 기법의 시간 복잡도는 다음과 같다.

제안된 기법은 권한에 대한 정보를 미리 계산하여 저장하여 둔 명시적 접근 권한 트리를 이용하므로 뷰를 생성하는 시간 복잡도는 최선의 경우 1, 최악의 경우가 n이 된다. 이와 같은 결과가 나오는 이유는 최 상위 노드가 부정으로 지정되는 경우(최선의 경우)와 전체 노드의 권한이 긍정으로 되는 경우(최악의 경우)를 가지기 때문이다. 그리고 제안된 기법은 접근 권한 트리를 생성하는 시간이 추가로 걸리는데 이 시간은 $2n$이다. 이유는 모든 권한 정보를 이용하여 만들어 내는 접근 권한 트리의 생성 알고리즘은 권한을 명시적으로 부여하는 시간과 부여한 권한간의 충돌 유무를 판별하는데 걸리는 시간 등 두 번의 전체 트리 탐색 시간이 소요되기 때문이다. 그러나 접근 권한 트리 생성은 미리 한번만 하고 뷰 생성에는 트리 정보를 사용만 하기 때문에 별도 시간이 필요없다.

위의 수치만으로 볼 때 관련 연구에서 제시한 Damiani 모델과 제안된 기법의 뷰 생성에 걸리는 시간 복잡도는 비슷하다고 생각 할 수 있으나, 응용분야의 실제적인 적용에 있어서 제안된 기법은 Damiani 모델보다 뷰 계산 시간을 현저히 감소시킬 수 있다. 이유는 권한에 관한 권한 정책은 사용자가 뷰를 요청할 때 마다 매번 바뀌는 것이 아니기 때문이다. 한번 정해진 정책은 자주 바뀌지 않으므로 권한 정보를 미리 계산하여 저장한 접근 권한 트리를 사용하였을 경우 사용자가 뷰를 요청했을 때, Damiani 모델은 평균 $\frac{5n}{2}$ 만큼의 시간이 걸리지만, 제안된 기법은 평균 $\frac{n}{2}$ 만큼의 시간이 걸린다. 이상에서 볼 수 있듯이, 제안된 기법은 사용자의 요청에 대해 Damiani모델에 비하여 빠른 뷰를 제공한다.

다음으로 제안된 기법에 대한 저장 공간의 오버헤드는 다음과 같다. 먼저, n을 권한 정보를 표현하는데 필요한 XML DTD의 노드 수라고 가정한다. 제안된 기법에서는 모든 구성요소가 권한 정보를 가지기 때문에 권한 정보를 표현하는데 필요한 XML DTD의 노드 수가 최소한 XML 문서를 구성하고 있는 전체 노드의 수 이상 존재 하게 된다. 그러므로 제안된 기법의 공간 복잡도는 $O(n)$이 된다. 이상에서 볼 수 있듯이 저장 공간 복잡도가 $O(1)$인 Damiani 모델에 비하여 제안된 기법은 더 많은 저장 공간의 오버헤드를 갖는다.

5. 결론

본 논문에서는 XML 문서의 미세 접근 제어를 위한 명시적 권한 기법을 제안하였다. 명시적 권한 정보는 접근 권한 트리 모델을 사용하여 저장한다. 현재 XML의 권한 접근 기법 중 많은 연구가 이루어지고 있는 묵시적 권한 기법은 상위 구성요소에 한 번의 권한 부여로 하위 구성요소에 권한을 부여하는 효과를 가지는 장점을 가지지만, 권한을 가지는 구성요소에 접근할 경우나 권한을 추가로 부여하는 경우 상위 구성요소와의 충돌의 유무를 판별하는 권한 체크시간의 오버헤드가 크다는 단점을 가지고 있다. 반면, 명시적 권한 기법은 각각의 구성요소 마다 권한 정보를 저장하는 기법으로 묵시적 권한 기법에 비하여 저장 공간의 오버헤드가 있지만, 각 구성 요소가 권한 정보를 가지고 있으므로, 즉각적인 접근 제어를 할 수 있고, 묵시적 권한 기법보다 권한 체크 시간이 작으므로, 권한 체크가 자주 발생하는 응용분야에 적용하면 권한 체크 시간을 현저히 감소시킬 수 있다.

전자 상거래나 병원 관리 등의 응용 분야와 같이 문서에 대한 권한 정책이 자주 수정되지 않고, 읽기에 대한 비율이 쓰기에 대한 비율보다 현저하게 많은 응용분야에 제안된 기법을 사용하면, 문서를 요청한 사용자에게 Damiani모델에 비하여 빠른 뷰를 제공한다. 하지만, Damiani 모델에 비하여 저장 공간의 오버헤드를 가지나 모든 XML 문서에 대해 권한 정보를 저장하는 것이 아니라 XML 문서의 DTD에 대한 권한 정보만 저장하므로 큰 문제는 아니라고 생각된다.

참 고 문 헌

[1] M.Kudo nad S.Hada. "XML Document Security Based on Provisional Authorization" Proc. 7th ACM Conf. Computer and Communication Security, ACM Press, NewYork, pp.87-96, 2000

[2] J.Linn and M. Nystrom. "Attribute Certification:An Enabling Technology for Delegation and Role-Based Controls in Distributed Environments", Proc 4th ACM Workshop on Role-Based Access Control, ACM Press, NewYork, pp.121-130, 1999.

[3] E.Bertino, S.Castano, and E.Ferriari. "Securing XML Documents with Author-X", IEEE Internet Computing, Vol.5, No.3, pp.21-31, 2001.

[4] A. Gabillon and E. Bruno. "Regulating Access to XML Documents", Proc. 15th Ann IFIP WG Working Conf, Database and Application Security, Kluwer Academic, Boston, pp.311-328,

2001.

[5] E Damiani et al "Securing XML Documents", Proc. 2000 Internet Conf, Extending Database Technology(EDBT2000), pp 121-135, 2000

[6] S Jajodia et al. "Flexible support for Multiple Access Control Policies", to be published in ACM Trans Database Systems, 2001

[7] C. Nlioudis, G. Pangalos and A. Vakal-i. "Security Model or XML data", Proc 2th International Conference on Internet Computing, 2001

[8] E. Damiani, S D C di Vimercati, S Paraboschi, P Samarati. "A fine grained access control system for XML documents", ACM Transaction on Information and System Security, Vol.5 No 2, pp.169-202, 2002.

[9] E Damiani, S. D C di Vimercati, S. Paraboschi, P. Samarati. "Controlling access to XML documents", IEEE Internet Computing, pp.18-28, 2001.

[10] Ravi Sandhu, Pierangela Samarati "Access Control: Principles and Practice", IEEE Commu nications Magazine, 1994.

[11] Bertino, Castano, Ferrari, Mesiti. "Controlled Access and Dissemination of XML Document", WIDM, 1999

[12] H He and R K Wong. "A role-based access control model for XML repositories", Proc. 1st Int Conf. on Webinfo Syst. Eng., Wise, HongKong, 2000.

[13] P Samarati,E. Damiani, S D. C di Vimercati, S Paraboschi "Design and implementation of an access control processor for XML documents", Comput. Netw, pp 59-75, 2000.

[14] E Bertino, S.Castano, and E.Ferrian "Specifying and enforcing access control policies for XML document sources, World Wide Web J.3,3, 2000

[15] T Bray et al "Extensible Markup Language(XML) 1.0", W3C, http:// www.w3c org/TR/ REC-xml, 2000.

[16] M. Bartel et al "XML-Signature Syntax and Processing", W3C, http:// www.w3c.org/TR/ xmlenc-core, 2000

[17] W3C "XML Path Language(XPath)", http:// www.w3c org/TR/xpath20, 2000.

[18] W3C "XML Access Sheet(XAS)", http:// www w3c org/TR/xas, 2000.

[19] W3C. "Document Object Model(DOM)", http:// www w3c.org/TR/REC-DOM- Level-1, 1998

[20] W3C. "XML schema", http://www w3c. org/TR/xmlschema, 2001.

[21] Bertino, E., "Data Hiding and Security in Object-Oriented Database," *In Proc. Int'l Conf on Data Engineering*, Tempe, Arizona, pp 338-347, Feb. 1992.

[22] Fernandez, E. B, Gudes, E., and Song, H, "A Model for Evaluation and Administ비율n of Security in Object-Oriented Database," *IEEE Trans. on Knowledge and Data Engineering*, Vol.6, No.2, pp.275-292, 1994.

[23] Bertino, E, Samarati, P., and Jajodia, S., "An Extended Authorization Model for Relational Databases," *IEEE Trans. on Knowledge and Data Engineering*, Vol.9, No 1, pp 85-101, 1997.

[24] Rabitti, F *et al.*, "A Model of Authorization for Next-Gene비율n Database Systems," *ACM Trans on Database Systems*, Vol 16, No 1, pp.88-131, 1991.

[25] Fernandez, E. B., Larrondo-Petie, M. M., and Gudes, E, "A Method-Based Authorization Model for Object-Oriented Databases," *In Proc. OOPSLA93 Conf. Workshop on Security for Object-Oriented System*, Washington D C, pp.135-150, 1993