XML 문서를 위한 묵시적 권한부여 기법

강정모[U] , 이헌길
강원대학교 컴퓨터·정보통신공학과
boo95002@hanmail.net , hglee@kangwon.ac.kr

An Implicit Authorization Technique for an XML Document

Jung-Mo Kang[U] Heon-Guil Lee
Dept. of Computer and Information, Kangwon National University

요 약

XML은 인터넷상에서 복잡한 문서의 원활한 처리와 신속한 탐색 및 항해가 가능한 차세대 웹 언어로 각광받고 있다. XML로 표현된 문서들은 세분화된 계층구조(granularity hierarchy)로 나타낼 수 있으므로 필요한 구성 요소에만 액세스 제어가 가능하다는 장점이 있다. 묵시적 권한 부여 기법은 명시적으로 저장된 권한으로부터 유도되는 권한기법으로 모든 구성 요소들에 대해 규칙들을 명시적으로 저장해야 하는 비효율적인 명시적 권한부여 기법보다 상위 구성 요소에 대한 한번의 권한 부여로 하위 구성 요소들에 동일한 권한부여 효과를 얻을 수 있다. 본 논문은 XML 문서를 위한 묵시적 권한 부여 기법을 제시하여 XML 문서의 액세스 제어 시 권한 부여 시간 및 메모리의 효율성을 높인다.

1. 서 론

정보화 사회로 발전해감에 따라 시스템 및 응용에 독립적인 문서정보가 요구되고, 문서 정보 교환, 검색 등의 처리를 위한 표준이 필요하게 되었다. 이에 웹 문서 양식인 XML이 등장하게 되었는데, XML(eXtensible Markup Language)은 SGML의 부분집합으로서 W3C(World-Wide Web Consortium)에 의해 제안된 확장성 마크업 언어이다 [7,9]. '확장성'이라는 말에서 알 수 있듯이 문서의 내용에 관련된 태그를 사용자가 직접 정의할 수 있으며, 그 태그를 다른 사람들이 사용할 수도 있다. 이런 이유로 XML을 구조적인 자료로 구성된 문서를 위한 마크업 언어라고도 한다.

현재 XML은 인터넷 상의 EDI나 전자상거래 등의 응용들로 적용이 확대되고 있는데 [7,8,9], XML 문서가 타인에 의해 쉽게 도청이나 조작되거나 오용된다면 문서에 대한 신뢰성이 떨어져 그 이용이 제한될 것이다. 이에 따라, XML 문서를 보호하기 위한 여러 보안 기법들이 제안되고 있다 [8,12].

본 논문에서는 저장된 XML 문서의 액세스 제어를 위한 묵시적 권한부여 기법을 제안한다. 명시적 권한 부여 기법은 모든 구성 요소에 대해 규칙들을 명시적으로 저장하는 것으로서 구현이 단순한 반면 상당히 비효율적이다 [2,5]. 하지만, 묵시적 권한부여 기법은 명시적으로 저장된 권한으로부터 유도되는 권한이므로 메모리를 감소시킬 수 있다 [2,5]. 묵시적 권한부여 기법을 사용하면 상위 구성 요소에 대한 한번의 권한 부여로 하위 구성 요소들에 동일한 권한 부여 효과를 얻을 수 있어 권한 부여 시간을 줄일 수 있다.

본 논문의 구성은 다음과 같다. 2장에서는 묵시적 권한 부여 기법에 대하여 설명하고, 3장에서는 XML 문서를 위한 제안된 묵시적 권한 부여 기법에 대하여 기술한다. 4장에서는 제안된 기법을 분석하고 평가하며, 마지막으로 5장에서는 결론 및 향후 연구 방향을 제시한다.

2. 묵시적 권한부여 기법

권한부여는 명시적 권한부여와 묵시적 권한부여로 나눌 수 있다. 명시적 권한부여 기법은 모든 구성 요소들에 권한을 명시적으로 저장하는 것으로서 구현이 단순한 반면에 상당히 비효율적이다. 묵시적 권한 부여 기법은 명시적으로 저장된 권한으로부터 유도되는 권한으로 유도를 결정하기 위한 계산 오버헤드를 수반하지만 메모리의 절약과 권한 부여 시간을 단축할 수 있다 [6]. 묵시적 권한부여는 긍정적 권한부여와 부정적 권한부여로 구별할 수 있다 [1,3,4]. 묵시적 권한 부여 기법에서는 명시적으로 권한이 부여되기 전에는 한 주체는 어느 객체에 대해서도 권한을 갖지 못함을 가정한다. 한 객체에 대해 권한을 가지지 못한다는 것은 주체가 그 객체에 접근할 수 없다는 의미이다. 부정적 권한부여의 개념은 긍정적 권한 부여를 보장한다. 즉, 주체가 객체에 대해 권한을 가지고 있지 않거나 부정적 권한을 가지고 있으면 객체에 대한 그 주체의 접근 시도는 거부된다. 실제로 현재 존재하는 권한부여 모델에서는 권한의 부재가 부정적 권한부여를 대

이 연구는 정보통신부 정보통신연구원 우수대학원 지원사업 과제로 수행된 것임

신하고 있다. 권한 부여는 강성과 약성 권한부여로도 구별할 수 있는데, 강성 권한 부여는 그 권한과 그 권한에 유도되는 모든 권한들에 대해 덮어쓰기를 허용하지 않는다. 반면에, 약성 권한 부여에 의해 유도된 권한들은 다른 권한부여에 의해 덮어쓰기를 허용한다 [1,3,4].

3. XML문서를 위한 묵시적 권한부여 기법

3.1 DOM 모델과 묵시적 권한부여 기법

DOM(Document Object Model)은 XML로 표현된 문서의 객체 모델로서 트리 구조로 XML 문서를 표현한다 [10]. 그림1은 우리가 흔히 볼 수 있는 구매 주문서를 XML의 형식으로 표현한 것이다.구매 주문서에서 보듯이 구매 주문서 내 신용카드 정보와 같은 중요 정보는 노출되면 개인에게 피해를 줄 수 있으므로 그에 대한 권한은 특정인만이 가져야 한다. 따라서, 특정 구성 요소 각각에 대해 권한 부여를 할 수 있는 방법이 필요하다.

```
<?xml version="1.0" encoding="EUC-KR"?>
<주문서 SOURCE="web" 고객타입="소비자" 화폐단위="천원">
  <고객>
    <성명> 홍길동 </성명>
    <주소> 강원도 춘천시 효자동 </주소>
  </고객>
  <주문품목록>
    <품목1>
      <부품명> 메인보드 </부품명>
      <수량> 1 </수량>
      <단가> 200 </단가>
    </품목1>
  </주문품목록>
  <결제방법>
    <신용카드>
      <소유자> 홍길동 </소유자>
      <카드번호> 4128240012301234 </카드번호>
      <유효기간> 2002 1/12 </유효기간>
    </신용카드>
  </결제방법>
</주문서>
```

그림 1. 구매주문서의 XML 표현

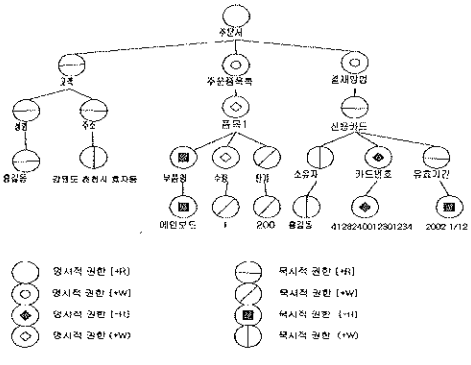

그림 2. 구매주문서에 대한 DOM의 트리구조

그림2는 그림1의 XML로 표현된 구매주문서를 DOM 모델로 표현한 것으로 각 노드들의 권한부여 예를 보여준다. 그림 2에서 보듯이 XML 문서를 DOM으로 나타내면 세분화된 계층구조로 표현되어 각 구성 요소 단위로 액세스 제어가 가능하다. 본 논문에서는 DOM으로 표현된 XML 문서에 대해 묵시적 권한부여 기법을 제안한다. 본 논문에서 제시한 묵시적 권한 부여 기법은 Rabitti이 제안한 권한부여 원칙을 적용한다 [4]. 첫째는 **일관성**으로 권한 강도가 같은 권한 집합 내에 각각 적용되는 성질로서, 어느 한 명시적 혹은 묵시적 권한의 타입이 다른 명시적 혹은 묵시적 권한의 타입의 부정이 될 수 없다. 둘째는 **비중복성**으로 강성 권한 집합 내에만 적용되는 성질로서, 임의의 명시적 강성 권한과 그에 의해 유도되는 묵시적 강성 권한이 권한 집합 내에 같이 존재할 수 없다. 마지막은 **공존성**으로 권한 강도가 다른 권한 집합간의 관계에 적용되는 성질로서, 임의의 한 노드에 명시적 약성 권한이 부여되어 있는 상태에서 그 노드에 강성 권한이 명시적 혹은 묵시적으로 부여될 때, 어느 한 권한의 타입이 다른 권한의 타입의 부정이 아닌 경우에 한하여 두 권한의 타입은 같이 존재할 수 있다. 이 세 가지 원칙을 이용하여 권한 부여시 권한 충돌 여부를 판정할 수 있는 호환성 행렬을 표1,2,3,4와 같이 구할 수 있다.

기존 \ 추가	(R)	(W)	(-R)	(-W)
(R)	F	T	F	T
(W)	F	F	F	F
(-R)	F	F	F	F
(-W)	T	F	T	F

표 1. 강성-강성 권한 호환성 행렬

기존 \ 추가	(R)	(W)	(-R)	(-W)
[R]	T	T	F	T
[W]	T	T	F	F
[-R]	F	F	T	T
[-W]	T	F	T	T

표 2. 약성-강성 권한 호환성 행렬

기존 \ 추가	[R]	[W]	[-R]	[-W]
(R)	F	T	F	T
(W)	F	F	F	F
(-R)	F	F	F	F
(-W)	T	F	T	F

표 3. 강성-약성 권한 호환성 행렬

기존 \ 추가	[R]	[W]	[-R]	[-W]
[R]	F	T	F	T
[W]	F	F	F	F
[-R]	F	F	F	F
[-W]	T	F	T	F

표 4. 약성-약성 권한 호환성 행렬

표1,2,3,4에서 제안된 묵시적 권한 호환성 행렬은 약성 권한부여 개념을 필요로 하며, 약성 권한 부여 개념은 또한 부정적 권한부여 개념을 필요로 한다. 이 개념은 명시적인 권한 부여가 없는 주체는 객체에 대한 어떠한 권한도 없다는 가정에 기초하고 있다. 표1,2,3,4에서 사용된 기호의 의미는 다음과 같다. ( )는 강성 권한을 [ ]는 약성 권한을 의미하며, - 표시는 부정 권한을 의미한다. R은 읽기 권한이며, W는 쓰기 권한, -R은 읽기 부정 권한이며, -W는 쓰기 부정 권한이다. T(True)는 권한 충돌이 없음을 F(False)는

권한 충돌이 있음을 나타낸다.

3.2 권한충돌 알고리즘

기존의 권한에 새로운 권한을 부여할 때 권한의 충돌 여부를 판정해야 한다. 묵시적 권한 부여 시 충돌 판정을 위해서는 권한을 부여하고자 하는 노드로부터 역으로 상위 노드로 거슬러 올라가는 방법으로 충돌을 판정한다. 그림3은 권한 충돌을 판정하는 알고리즘을 기술한 것이고, 그림4는 DOM의 특정 노드를 액세스할 때 액세스가 가능한가를 판별하는 알고리즘이다. 시스템에서의 권한은 기본적으로 3-tuple (s, o, a)로 정의한다. 여기서, s, o, a는 다음과 같다.

- s: 권한에 대한 주체(authorization subject)의 집합인 S의 한 원소 (사용자 혹은 사용자 그룹 등)
- o: 권한의 객체(authorization object)의 집합인 O의 한 원소 (DOM의 노드, 즉 텍스트, 엘리먼트, 엔티티 등)
- a: 권한의 타입(authorization type)의 집합인 A의 한 원소 (읽기, 쓰기 등)

그림3과 그림4의 알고리즘에서 사용한 기호는 다음의 의미를 갖는다.

- n : 권한을 부여하거나 액세스하고자 하는 노드에 대한 포인터
- (s_n, n, a_n) : 노드 n에 부여하거나 액세스할 권한
- p : 부모 노드에 대한 포인터
- (s_p, p, a_p) : 노드 p의 명시적 권한
- parent(n) : 노드 n의 부모 노드를 반환하는 함수

```
algorithm Conflict (n, (sn, n, an))
begin
1. p = parent(n)
2. 명시적 권한이 부여된 노드를 찾거나 최상위 노드까지 다음을 수행한다.
  2.1 p = parent(p)
3. 만약 p가 최상위 노드이고 명시적 권한이 없으면 p의 권한 (sp, p, ap)를 모든
   경우 부정으로 설정한다.
4. 표 1,2,3,4에 의해 p의 권한 (sp, p, ap)와 n에 부여할 권한 (sn, n, an)의 충
   돌 여부를 판정한다.
end
```

그림 3. 권한 충돌 알고리즘

```
algorithm AccessControl(n, (sn, n, an))
begin
1. p = n
2. 명시적 권한이 부여된 노드를 찾거나 최상위 노드까지 다음을 수행한다.
  2.1 p = parent(p)
3. 만약 p가 최상위 노드이고 명시적 권한이 없으면 p의 권한 (sp, p, ap)를 모든
   경우 부정으로 설정한다.
4. p의 권한 (sp, p, ap)와 액세스하고자 하는 권한 (sn, n, an)를 비교한
   후 노드 n에 대한 액세스 가능 여부를 결정한다.
end
```

그림 4. 액세스 제어 알고리즘

4. 분석 및 평가

본 논문에서 제시한 묵시적 권한 부여 기법을 사용하면 XML 문서의 액세스 제어를 문서를 구성하는 각 구성 요소 레벨에서 표현할 수 있다. 제안된 묵시적 권한 부여 기법은 모든 구성 요소에 대해 규칙을 기술하는 명시적 권한 부여 기법에 비해 메모리를 절약한다. 또한, 상위 구성 요소에 대한 권한 부여가 하위 구성 요소들에 동일한 권한을 자동적으로 부여함으로써 권한 부여 시간을 감소시킬 수 있다. 그러나, 권한 부여 시 권한 충돌을 탐지하는 계산과 특정 구성 요소에 대한 액세스 가능 여부를 계산해야 하는 오버헤드가 있다. 그림 3과 그림 4의 알고리즘의 시간 복잡도(time complexity)는 DOM 트리의 노드 수가 n일 때 최악의 경우 $n-1$ 이지만 보통 DOM 트리가 균형을 이룬다고 가정하면 평균적으로 $\log n$ 이 된다. 따라서, 계산 시 고려해야 할 내부 노드의 개수는 전체 노드에 비해 상대적으로 적으므로 계산 시간의 오버헤드가 그렇게 심각하지는 않다.

5. 결론 및 향후 연구과제

본 논문에서는 XML 문서의 액세스 제어를 위한 묵시적 권한 부여 기법을 제안하였다. 제안된 기법은 XML 문서의 각 구성 요소 레벨에서 액세스 제어를 가능하게 한다. 그리고, 명시적 권한 부여 기법에 비해 메모리의 절약과 권한 부여 시간을 감소시킨다. 하지만, 권한 부여 시 권한 충돌 여부를 판정하고, 구성 요소 액세스 시 액세스 가능 여부를 검사하는 계산 오버헤드가 있다. 향후 충돌 여부를 판정하는 오버헤드와 액세스 시 검사 오버헤드를 감소시킬 수 있는 기법에 대한 연구가 필요하다.

6. 참고 문헌

[1] 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.

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

[3] 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.

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

[5] Fernandez, E. B., Larrondo-Perie, 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, Sep. 1993.

[6] 손태종, 조완섭, 황규영, "객체지향 데이터베이스 시스템에서 계승을 통한 묵시적 권한부여 기법의 특성 분석," *한국정보과학회 가을 학술발표논문집*, vol.23, no.1, pp.131-134, 1996.

[7] David Hunter, *XML*, Information Publishing Group, Oct. 2000.

[8] IETF W3C, *XML-Signature Core Syntax and Processing*, Jan. 2000.

[9] W3C, *Extensible Markup Language (XML) Version 1.0*, http://www.w3.org/TR/REC-xml/, Feb. 1998.

[10] W3C, *Document Object Model Level1*, http://www.w3.org/TR/REC-DOM-Level-1/, 1998.