情報保護學會論文誌
第15券 第1號, 2005. 2

접근제어 정책구현을 위한 역할기반 XML 암호화*

최 동 희,[1†] 박 석[2‡]
[1]국방과학연구소, [2]서강대학교

Role based XML Encryption for Enforcing Access Control Policy

Dong-Hee Choi,[1†] Seog Park[2‡]
[1]Agency for Defense Development, [2]Sogang University

요 약

XML문서가 웹 문서의 표준으로 자리잡음에 따라 많은 정보들이 XML문서의 형식으로 표현되면서 XML문서의 보안에 관한 요구도 커지고 있다. XML문서의 보안은 암호화, 전자서명, 키 관리, 접근제어에 관한 연구가 활발히 이루어지고 있다. 현재까지의 XML보안에 관한 연구는 암호화와 전자서명 같은 통신상의 보안에 관한 연구가 중심이 되었으나, XML문서가 방대해지고 복잡해짐에 따라 XML문서에 대한 통신상의 보안 뿐 아니라 관리적인 보안이 필요하게 되었다. 이러한 관리적인 보안은 접근제어를 통해 보장할 수 있다. 하지만 XML문서의 암호화는 단순한 통신상의 보안만을 제공할 뿐 관리적 보안 요소인 다양한 사용자와 다양한 접근권한 정책을 반영하지 못하고 있다. 본 논문에서는 XML문서의 보안을 위해 역할별 권한정책을 반영한 접근제어를 보장하는 주체기반 암호화 기법을 제안한다. 접근제어를 보장하는 암호화를 수행함으로써 다양한 사용자의 다양한 권한정책을 반영할 수 있다. 또한 제안하는 암호화 기법은 전처리 과정을 이용하여 기존의 접근제어 기법의 복잡한 접근 권한 평가 비용을 줄일 수 있다.

ABSTRACT

As a large quantity of information is presented in XML format on the web, there are increasing demands for XML security. Research area of XML security is about Encryption, Digital Signature, Key management and Access Control. Until now research on XML security has been focused on the security of data network using digital signature and encryption technology. As XML data become extensive and complex XML security comes to involve not only network security but also managerial security. Managerial security is guaranteed through access control. But XML Encryption supports simple network security. So it can't support multiple users and multiple access control policies. In this paper, we propose an integration method of encryption and access control policy for securing XML documents. This methodology can support multiple authorizations of multiple users with integrating access control. And this can reduce the cost of evaluation process of the existing complicated access authorization with pre-processing.

Keywords : *XML security, encryption, access control*

접수일 : 2004년 8월 12일 ; 채택일 : 2004년 12월 20일
* 본 연구는 2004년도 서강대학교 교내연구비 지원에 의하여 이루어졌습니다.
† 주저자, fay17@add.re.kr
‡ 교신저자, spark@dblab.sogang.ac.kr

Ⅰ. 서 론

XML은 웹 환경에서 데이터를 표현하기 위한 마크업 언어이다. 이러한 XML은 SGML에서의 복잡성을 제거하고, HTML에서의 고정된 태그의 한계에서 벗어나 사용자가 문서 구조를 정의할 수 있다. 이와 같은 장점 때문에 W3C에서는 XML을 웹 데이터의 표준으로 제정했다. XML의 표준화 이후 많은 데이터가 XML로 표현되기 시작했고, 현재는 웹 상에 정부나 기업 등의 다양한 XML 데이터가 존재하고 있다. 웹 상에서 존재하는 XML문서의 정보는 네트워크를 통해 분산되고 공유되므로 XML문서의 정보의 유출을 막기 위한 기밀성 보장이 매우 중요한 보안의 요구사항으로 자리잡게 되었다. 기밀성의 보장은 많은 어플리케이션에서 중요한 부분이다. 예를 들어 의료기록을 살펴보면 인증과 관련된 문서의 각 정보는 다른 수준의 보안을 가질 필요가 있다. 병원의 직원 인사 기록은 철저한 보안을 요구하는 성과나 연봉 정보, 회사 내에서 빈번히 쓰이는 전화번호나 주소 같은 정보를 포함한다. 또한 환자정보는 훨씬 민감한 신용카드 번호가 포함되어 있을 것이다. 이런 각각의 항목은 다른 키로 암호화 되어야 하며, 각기 다른 계층의 병원 직원들로부터 보호되어야 한다. 또한, 기업의 보안 정책은 매우 복잡하게 구성되며 이들에 따른 다른 보안 요구사항을 만족하기 위해서는 접근제어가 이루어져야 한다. 이처럼 기업이나 정부의 복잡한 보안정책을 만족하기 위해서는 접근제어 정책을 반영한 암호화가 이루어져 한다.

이에 본 논문은 암호화와 접근제어를 통합함으로써 전송계층상의 보안뿐만 아니라 사용자의 차별적 접근을 허용하는 관리상의 보안 요구사항을 만족시키고자 한다. 암호화에 접근제어 정책을 반영하는 방법은 문서에 모든 권한을 반영하여 주체의 역할에 따른 접근권한에 따라 같은 역할에게 허용된 문서의 부분들을 하나의 키로 암호화한다. 이렇게 역할기반 암호화를 수행하게 되면 기존의 노드 단위 암호화로 발생했던 문제점인 키의 생성과 암호화 횟수를 단축할 수 있다. 또한 권한의 평가가 사용자가 요청한 후에 이루어지는 것이 아닌 암호화 수행과정에서 이루어지므로 기존의 접근제어의 복잡하고 반복적으로 수행되었던 권한 평가의 비용을 줄일 수 있다.

본 논문은 다음과 같이 구성되어 있다. 2장에서는 관련연구로 접근제어, 암호화, 접근제어와 암호화의 통합에 관해 살펴보고, 3장에서는 XML문서의 보안을 위한 요구사항과 역할기반 암호화기법의 제시와 수행과정을 살펴본다. 4장에서는 제안하는 암호화 기법을 기존의 기법과 비교하고 분석하며, 5장에서는 결론 및 추후연구에 대해 기술한다.

Ⅱ. 관련연구

1. W3C XML Encryption

W3C의 XML 암호화[1]는 다양한 사용자를 위한 암호화 방법을 몇 가지 제공한다. 가장 간단한 방법은 모든 사용자에게 XML문서의 동일한 부분을 볼 수 있도록 허용하는 것이다. 이 경우 값은 한번만 암호화 되고, 그 키는 각각의 사용자를 위해 다시 암호화 된다. 다양한 사용자를 위한 암호화의 다른 방법은 이미 암호화된 값을 다시 암호화하는 방법이 있다. 이 방법은 모든 사용자에게 제공되는 문서의 내용이 동일하지 않은 경우에 사용한다.

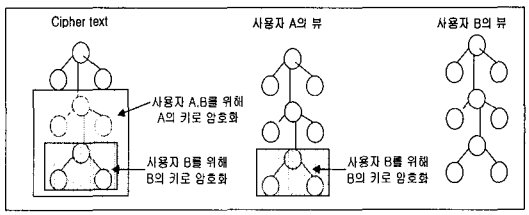


그림 1. 슈퍼 암호화

그림 1은 슈퍼 암호화를 그림으로 나타낸 것이다. 사용자 B를 위해 XML문서 트리의 일부분을 B의 키로 암호화 한다. 암호화 된 내용과 트리의 일부분을 사용자 A와 B를 위해 다시 A의 키로 암호화 한다. 사용자 A는 복호화했을 때 문서의 외부의 암호화된 부분만 복호화할 수 있고, B의 키로 암호화된 부분은 복호화할 수 없다. 사용자 B의 복호화는 두 단계로 이루어진다. 사용자 B는 A의 키와 B의 키 모두를 가진다. A의 키로 외부의 암호화된 부분을 복호화하고 다시 자신의 키로 내부의 암호화된 부분을 복호화하게 된다. 두 단계를 모두 거치면 사용자 B는 문서의 전체를 볼 수 있게 된다.

하지만 슈퍼암호화는 정책기반의 암호화가 아닌 단순한 기밀 정도에 따라 암호화를 수행하고 있으며, 키의 공유와 중첩 암호화로 인해 다양한 사용자의 차별적 접근에 따른 암호화를 올바르게 수행할 수 없는 단점을 가진다.

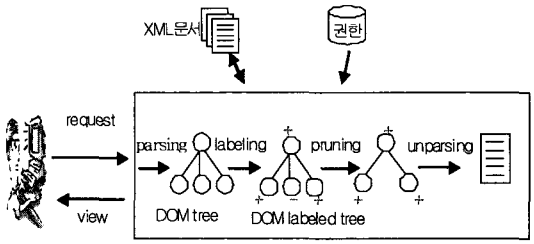


그림 2. XML문서의 접근제어 수행과정

2. XML문서의 접근제어

Damiani 등이 제안하는 접근제어 모델은 DOM 트리를 이용하여 XML문서와 DTD의 엘리먼트에 접근권한을 설정하고, 설정된 접근권한 정보에 의해 사용자의 XML문서의 접근을 제어한다.

그림 2는 XML문서의 접근제어 수행과정을 보여준다. 사용자가 XML문서를 요청하면 먼저 DOM 트리를 얻기 위한 XML문서의 파싱(parsing)을 수행한다. 파싱 수행 후 XML문서와 권한 데이터베이스의 권한을 참조하여 DOM 트리의 노드에 접근 허가(+) 혹은 거부(-)를 의미하는 sign 값을 설정하게 된다. 이렇게 DOM 트리의 노드에 권한을 설정하는 작업을 레이블링(labeling)이라 한다. 최종적으로 레이블링된 DOM 트리에서 sign 값이 -로 설정된 노드는 제거하고, +로 설정된 노드만을 XML 형식으로 사용자에게 반환한다. 이러한 접근제어 과정을 통해 사용자는 XML문서 중에서 자신이 볼 수 있는 권한이 있는 부분만을 접근하게 되므로, 비인증된 사용자로부터 데이터의 기밀성을 보장할 수 있다.

하지만 전체의 DOM 트리가 메모리 상에 적재되어야 하고, DOM 트리의 모든 노드에 접근 권한을 설정하기 위한 반복적인 트리의 검색으로 많은 메모리가 사용되며, 복잡한 권한의 평가 과정으로 인해 시스템의 성능 저하를 초래할 수 있는 문제점이 있다. 또한, 메모리의 DOM 트리에서 기밀정보를 포함한 모든 문서 전체를 처리하기 때문에 data diddling문제가 발생할 수 있고, 네트워크를 통해 기밀정보가 포함된 문서를 전달하므로 packet sniffing과 같은 정보의 유출 문제가 발생할 수 있다.

3. XML문서의 암호화와 접근제어의 통합모델

XML Pool Encryption[6]는 XML문서의 암호화는 다양한 사용자를 위한 암호화 기법을 제공하지

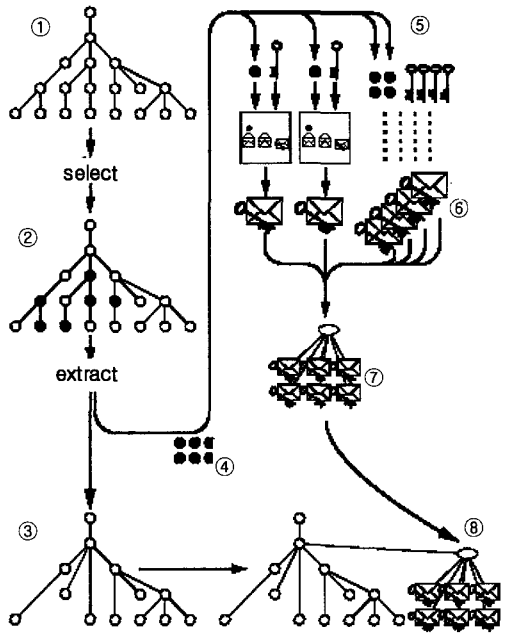


그림 3. XML Pool Encryption의 수행과정[6]

않다는 것에 착안하여 XML문서의 암호화에 접근제어를 도입하여 다른 권한을 가진 다양한 사용자를 위한 암호화 기법을 제시하고 있다. 접근제어를 적용한 방법은 XML문서의 각 노드(각 엘리먼트, 애트리뷰트) 중에서 기밀 정보를 가진 노드를 암호화한다. 각 기밀 노드는 다른 키로 암호화하여 암호화 풀(pool)에 저장하게 된다. 사용자의 요청이 들어오면 기밀 노드 중에서 사용자에게 권한이 있는 노드만을 선별하여 키와 함께 전달하게 된다. 아래 그림 3은 XML Pool Encryption의 수행과정을 보여준다.

위의 그림을 보면 ①~②번 수행과정은 전체 XML문서에서 기밀정보를 추출하는 과정이고, ③은 기밀정보를 제외한 문서이다. ④번처럼 암호화된 노드는 풀에서 관리하게 된다. 문서의 요청이 들어오면 권한을 확인하여 암호화된 노드의 키와 함께 사용자에게 전달되게 된다.

하지만 XML Pool Encryption은 요청한 사용자에게 권한이 존재하는 노드들만을 전달하기 위해서 각 노드마다 권한이 존재하는 사용자들의 집합을 유지하고 있어야 한다. 만약 문서의 대부분이 다른 권한을 가진 다양한 사용자에 대한 기밀정보라면 문서의 대부분의 노드를 각각 다른 키로 암호화해야 한다. 이 경우 키가 무수히 많이 필요하게 되고, 각 기밀노드의 키의 분배 문제와 암호화, 복호화를 위

한 처리 능력도 높아야 한다. 또한 암호화된 노드들을 분리하여 풀에 저장하므로 각 노드에 원래의 위치를 찾기 위해 트리에서의 위치정보를 가지고 있어야 하고, 자신의 위치정보 뿐만 아니라 다른 노드의 위치정보를 확인해야만 자신의 위치를 찾을 수 있다.

Ⅲ. 역할기반 암호화를 통한 XML 접근제어

본 논문은 접근제어의 권한 정책을 반영한 XML 문서의 암호화를 수행하는 것이다. 제안하는 기법을 기술하기에 앞서 다음과 같은 환경을 가정한다.

검색연산이 빈번히 발생한다.

기술된 권한정책의 변경이 빈번히 발생하지 않는다.

다양한 사용자와 다양한 수준의 기밀정보가 존재한다.

1. 역할기반 암호화를 통한 XML 접근제어의 구조

접근제어는 인증을 거친 사용자가 원하는 자원을 요청하면 시스템은 사용자의 권한을 확인하고 요청한 자료를 제공할 것이지 아닌지를 결정하게 된다. 제안하는 역할기반 암호화 기법의 구조는 그림 4와 같다.

문서가 생성되면 pre-processing processor에 의해 전처리 과정이 수행된다. 전처리 과정은 XML 문서와 권한 데이터베이스를 이용하여 대체정보(RI)를 생성하고, 문서를 암호화하여 저장하고, 생성된 키를 key server에 전달한다.

사용자는 시스템에 인증을 거친 후 역할을 할당받게 된다. 인증된 사용자가 문서를 요청하면 security processor에 의해 대체정보를 확인하게 된다. 사용자에게 허용된 권한이 존재하면 representation processor는 대체정보와 암호화된 문서 중 권한이 있는 부분, 키 서버에서 역할에 해당하는 키 집합을 받아 사용자에게 전달할 문서를 재구성 한다. 구성된 문서는 사용자에게 전달되게 되고 사용자는 복화화하여 자신에게 권한이 존재하는 문서의 부분만을 볼 수 있게 된다.

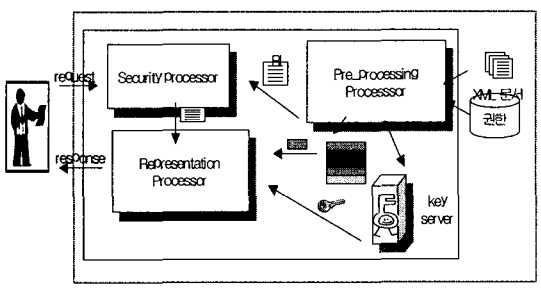


그림 4. 제안하는 역할기반 암호화 기법의 구조

2. 접근권한

접근권한은 다음과 같이 정의한다.

정의 1 (접근권한) 권한 a∈권한 은 다음의 다섯개의 튜플을 가진다.

〈subject, object, action, sign, type〉

- □ subject ∈ 역할 는 권한이 주어지는 주체로 사용자의 역할을 의미한다.
- □ object는 권한이 주어지는 객체로 path expression으로 표현한다.
- □ action = read 는 접근권한을 의미하며, 읽기 연산으로 제한한다.
- □ sign ∈{+, -}은 권한의 부호를 의미하며, +는 허용을 -는 거부를 의미한다.
- □ type ∈{L, R}은 권한의 타입을 의미하며, L(Local)은 권한의 전파가 기술된 엘리먼트와 그 어트리뷰트에 적용되는 것을 의미하며, R(Recursive)는 권한의 전파가 기술된 엘리먼트의 모든 하위 엘리먼트와 어트리뷰트에 적용됨을 의미한다.

정의된 접근권한은 사용자의 역할에 따라 기술된다. 역할단위의 권한은 개인이 아닌 역할에 권한이 기술되므로 권한과 객체정보가 변경되어도 역할에 기술된 권한만 변경하면 된다. 또한 역할에 따른 권한의 상속을 허용하므로 권한의 기술이 용이하다.

3. 전처리 과정의 수행

앞에서 정의된 접근권한을 바탕으로 전처리 과정을 수행하게 된다. 전처리 과정은 문서의 생성시에 수행된다.

그림 5는 전처리 과정의 수행을 나타낸다. 전처리 과정은 XML문서에 권한정책을 반영하여 권한이 존재하는 주체의 역할에 따라 암호화를 수행하고 대체정보를 생성하는 과정이다. 문서는 권한이 있는 주체의 역할에 따라 문서의 부분이 하나의 키로 암호화 되고 암호화된 문서의 일부는 원래의 문서와 분

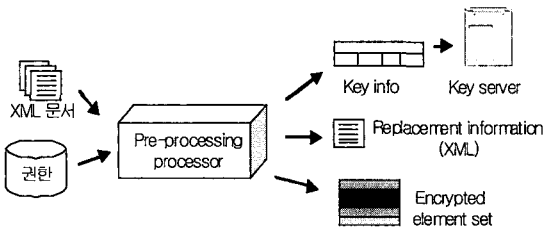


그림 5. 전처리 과정의 수행

리된 암호화 집합에 저장되게 되고 암호화된 노드가 있었던 문서의 위치에는 대체정보가 놓여진다.

3.1. 레이블링의 수행

XML문서와 권한 데이터베이스를 참조하여 pre-processing processor는 문서의 트리 노드에 주체에 접근이 허용되는지 거부되는지를 표기하는 레이블링을 수행한다. 표기되는 레이블링은 〈subject, sign〉으로 구성된다. Subject는 접근권한의 주체의 역할을 의미하며 sign은 접근권한의 허용, 거부를 의미한다. 초기에 표기되는 레이블은 동일한 subject의 역할에 대해 여러 개의 레이블이 존재할 수 있다. 이와 같이 동일한 subject 역할에 하나이상의 권한이 기술되고 이 권한들의 sign값이 같지 않을 때를 충돌(conflict)이 발생했다고 한다. 충돌은 most specific object takes precedence 원리와 closed policy 원리에 따라 해결한다. 즉, 전파되어 발생한 권한보다는 직접 기술된 권한이, DTD에 기술된 권한보다는 문서에 직접 기술된 권한이 우선순위를 갖게 되고, 거부권한이 우선순위를 가지게 된다. 초기 레이블링을 바탕으로 충돌을 해결하면 동일한 subject에 대해 레이블은 하나만 존재하게 된다. 이를 바탕으로 암호화를 수행하고 대체정보를 생성하게 된다.

3.2. 암호화와 대체정보의 생성

DOM 트리에 기술된 최종 레이블을 바탕으로 pre-processing processor는 암호화를 수행하고 대체정보를 생성한다. 대체정보는 다음과 같이 기술한다.

정의 2 (대체정보) 대체정보는 다음의 엘리먼트와 어트리뷰트로 구성된다.

〈!ELEMENT 문서의 임의 엘리먼트 (unit)*〉
〈!ATTLIST unit subject CDATA #REAUIRED sign (+ |-) #REQUIRED e_id ID #IMPLIED path CDATA #REQUIRED〉

대체정보는 동일 레이블 집합을 소유한 노드들마다 하나의 unit 엘리먼트를 구성하게 된다. 레이블이 시작되는 노드의 하위에 노드들은 모두 암호화되어 다른 곳에 저장되게 되고 그 위치에 unit 엘리먼트가 위치하게 된다. unit 엘리먼트는 subject, sign, e_id, path의 어트리뷰트를 가지게 된다. subject와 sign은 레이블에 표기된 subject와 sign의 값을 가진다. e_id는 문서 중 암호화된 부분을 식별하기 위한 것이다. 사용자의 요청이 들어오면 대체정보의 e_id 값을 확인하여 암호화된 부분을 사용자에게 전달 시켜준다. 또한 암호화된 부분을 복호화하기 위한 키를 사용자에게 전달할 때도 e_id를 이용하게 된다. path는 unit의 부모 노드에서 레이블이 시작하는 노드까지의 path expression를 가지게 된다. unit의 부모 노드에서 시작되는 레이블은 path값은 널을 가지게 되고, unit의 부모노드보다 하위에서 시작하는 레이블은 unit의 부모노드부터 레이블이 시작되는 노드까지의 path 정보를 가지게 된다. 이 path정보는 사용자가 문서를 요청하여 복호화 한 후 문서가 재구성될 때 원래의 위치를 바르게 찾기 위해 필요하게 된다. 이와 같은 대체정보를 유지함에 따라 권한 데이터베이스의 검색 없이 접근제어가 가능하게 된다.

암호화는 대체정보를 바탕으로 수행된다. 동일 subject의 역할에 대해 기술된 unit은 동일 키로 암호화가 수행되고 그 키는 해당 subject에 전달되게 된다. 만약 문서의 서브 트리가 여러 subject에 대해 레이블이 존재하면, 이 서브트리는 여러 사용자에게 접근권한이 존재하는 것으로 하나의 subject의 키로 암호화할 수 없다. 이 경우는 기술된 subject들이 역할계층구조상에서 포함관계가 존재하는지 확인한다. 만약 포함관계가 존재한다면 역할계층구조에서 하위 역할의 키로 암호화를 수행한다. 만약 포함관계가 존재하지 않는다면 기술된 어떤 subject의 키도 아닌 제 3의 키로 암호화를 수행해야 하고, 제 3의 키는 포함관계가 존재하지 않는 해당 subject에 모두 주어져야 한다. 이것은 키서버에서 관리, 수행하게 된다. 제 3의 키로 암호화를 수행하는 경우는 많이 발생하지 않는다. 왜냐하면

여러 subjct의 역할에 모두 허용이 된다면 역할계층구조 상에서 해당 subject를 포함하는 상위 역할이 존재할 것이며, 이 상위 역할의 키로 암호화를 수행하게 되기 때문이다. 암호화는 대칭키 암호화 기법으로 수행된다.

4. 키 서버

키 서버는 암호화의 키들이 저장되며, 사용자의 요청이 발생하면 사용자의 역할에 따라 알맞은 키를 제공해준다. 키 서버는 역할계층구조를 가지고 있어 사용자의 역할에 해당되는 키와 하위 역할의 키를 함께 제공한다. 이를 위해 키 서버는 역할 계층구조를 가지고 있어야 한다. 키 서버에 저장되어 있는 키는 어떤 사용자에게 전달할 것인지, 어떤 암호화된 부분에 관한 키인지에 관한 정보를 저장하고 있어야 한다. 키는 **〈e_id, key〉**로 구성된다. e_id는 암호화된 문서의 부분들을 식별하는 것으로 해당부분을 키를 이용해 복호화하면 된다.

키 서버는 사용자의 역할계층구조를 유지하고 있어야 한다. 왜냐하면 요청자는 자신의 역할에 기술된 권한 뿐만 아니라 자신의 역할이 포함되어 있는 하위 역할의 권한까지 가지게 되기 때문이다. 그러므로 요청자에게 키를 전달 시 자신이 속한 하위 역할의 키까지 포함해야 한다.

5. 주체기반 암호화 기법의 수행과정

이번 장에서는 제안하는 암호화 기법의 실제 XML문서의 전처리 과정과 사용자의 요청에 따른 전달 과정을 살펴본다. 그림 6은 제안하는 암호화 기법의 수행과정을 나타낸다.

전처리 과정의 수행은 다음의 단계로 구성된다.

1. 초기 레이블링(initial labeling) 단계 : 접근권한을 바탕으로 XML문서의 각 노드에 주체의 역할 각각에 대한 접근 허용, 거부를 표기한다.
2. 충돌의 해결(conflict resolution) 단계 : 동일 주체에 대해 허용과 거부가 모두 존재하는 경우 충돌 해결 원리에 따라 하나의 접근권한만을 결정한다.
3. 그룹핑(grouping) 단계 : 레이블을 바탕으로 동일 레이블이 존재하는 노드들을 그룹화를 수행한다.
4. 대체정보생성(create replacement information) 단계 : 그룹화된 노드별로 주체와 객체, 접근권한, 암호화 아이디 등의 권한정보를 대체정보로 생성한다.
5. 암호화(Encryption) 단계 : 대체정보를 바탕으로 그룹화된 노드들의 주체의 역할 키로 암호화 한다.

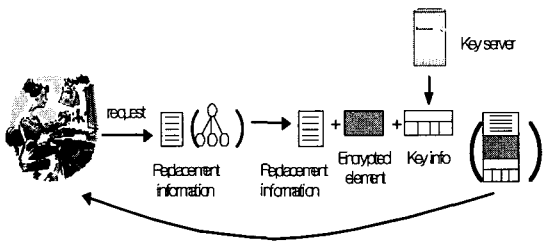


그림 6. 제안하는 암호화 기법의 수행과정

전처리 과정의 수행 후 인증을 거친 사용자가 해당 XML문서를 요청하게 되면 다음과 같은 과정이 수행된다.

1. 대체정보확인 단계 : 사용자의 역할을 바탕으로 해당문서에 접근권한이 존재하는지 대체정보를 확인한다.
2. 문서의 재구성 단계 : 만약 사용자에게 권한이 존재하면 수정된 대체정보와 사용자가 허용되는 암호화된 문서, 이를 복호화할 수 있는 주체의 키 집합을 재구성하여 사용자에게 전달한다.
3. 복호화 단계 : 사용자는 재구성된 문서를 복호화하여 자신에게 접근권한이 존재하는 문서의 부분만을 접근하게 된다.

본 논문에서 사용할 예제는 웹 기반의 의 환자정보 문서이다. Medical.xml의 그래프표현은 그림 7과 같다.

Medical.xml은 크게 네가지 정보로 구성된다. personal info는 환자의 신상에 관한 정보로 구성되고, Medical characteristic은 신체특성 정보를 포함하고 있다. billing_info는 지난 병원비 내역과 신용카드의 정보가 Medical history에는 현재까지 진료기록으로 구성되며 confidential과 sensitive의 두가지 type으로 구분할 수 있다. 병원시스템의 접근권한을 그림 8과 같다.

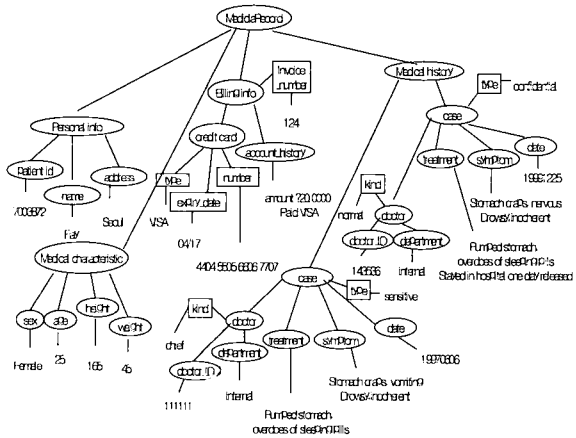


그림 7. medical.xml의 그래프 표현

```
<staff, Medical.xml:/MedicalRecord/per-
sonal_info, read, +, R>
<billing_staff, Medical.xml:/MedicalRe-
cord/biiling_info, read, +, R>
<doctor, Medical.xml:/MedicalRecord, read,
+, R>
<doctor, Medical.xml:/MedicalRecord/bill-
ing_info, read, -, R>
<doctor, Medical.xml:/MedicalRecord/Me-
dical_history/case[.@type="confidential"],
read, -, R>
<head_doctor, Medical.xml:/MedicalRe-
cord/Medical_history/case[.@type="confi-
dential"], read, +, R>
```

그림 8. 접근권한

기술된 접근권한의 의미는 다음과 같다. staff은 personal info의 하위 노드들을 읽을 수 있는 권한을 가진다. billing_staff은 billing_info의 하위 노드들을 읽을 수 있는 권한을 가진다. doctor는 MediclaRecord의 하위 노드들을 읽을 수 있는 권한을 가진다. 하지만 doctor는 billing_info와 Medical history의 type이 confidential이면 읽을 수 있는 권한이 없다. Head_doctor는 Medical history의 confidential type도 읽을 수 있다.

그림 9는 병원정보시스템의 역할계층구조를 보여준다. 수행하는 예제의 병원정보시스템의 역할은 staff, billin_staff, doctor, head_doctor로 가정한다. 모든 역할은 staff역할에 포함되게 된다. Head_doctor는 doctor 역할에 상위가 되어 doctor역할과 head_doctor의 역할을 수행하게 된다.

Medical.xml이 생성되면 전처리 과정을 수행하

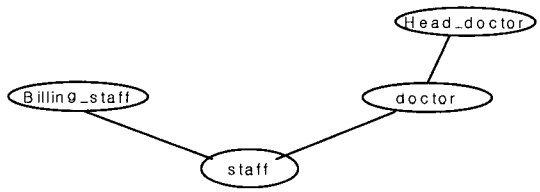


그림 9. 병원정보시스템의 역할계층구조

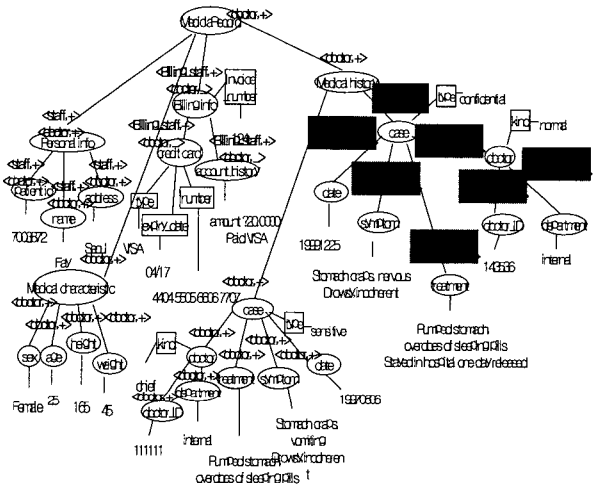


그림 10. 최종 레이블이 표기된 medical.xml

게 된다. 초기 레이블링 과정과 충돌을 해결한 결과는 그림 10과 같다.

최종 레이블이 결정되면 이를 바탕으로 대체정보를 생성하고 암호화를 수행해야 한다. 그림 11은 생성된 대체정보를 보여준다.

각 unit의 하위에 존재하던 노드들은 암호화되어 암호화 엘리먼트 집합에 e_id와 함께 저장된다. 만약 의사 '아영'이 환자정보문서 Medical.xml의 읽기 요청하면 시스템의 사용자의 역할을 확인하고

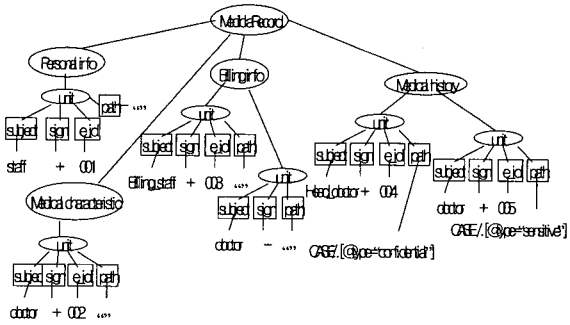


그림 11. medical.xml의 대체정보

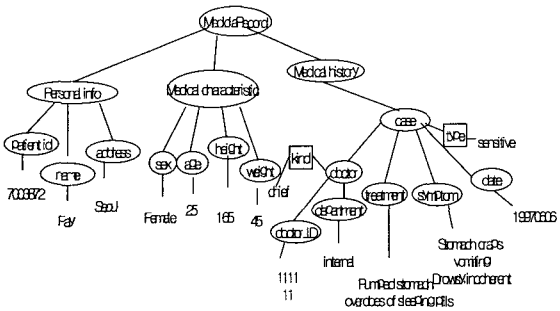


그림 12. 아영에게 보여지는 문서

Medical.xml의 대체정보를 확인하게 된다. 대체정보에 의사 역할 및 역할계층구조상에서 하위 역할에 권한이 있는 unit을 찾아 대체정보를 바탕으로 사용자에게 전달할 문서를 재구성하게 된다. 의사는 staff과 doctor에 허용된 부분을 볼 수 있다. '아영'에게는 재구성된 문서가 키 집합과 함께 전달된다. 키 집합 {(001,staff's key), (002,doctor's key), (005,doctor's key)}이 전달되게 된다. 사용자는 전달 받은 문서를 키 정보를 이용하여 복호화 하면 그림 12와 같다.

Ⅳ. 평가 및 비교 분석

제안하는 암호화기법은 접근제어와 암호화를 통합하여 주체 기반의 암호화를 수행하는 기법이다. 제안하는 기법의 평가는 기존의 암호화 기법과 비교하는 것이다. 비교의 기준은 접근제어와 암호화의 요구사항을 얼마나 만족하는가와 관리의 용이성이다. 비교의 대상은 관련연구에서 제시한 W3C XML Encryption[1]와 XML Pool Encryption[6]으로 수행한다. 비교의 결과는 표 1과 같다.

다양한 조직 단위 권한 지원은 조직 구조를 반영한 권한 기술을 가능하게 하므로 권한의 관리상의 용이성을 제공받게 된다. 제안하는 기법은 역할단위의 권한 기술로 권한 기술이 용이하며 권한의 상속을 허용한다.

fine-grained access control은 엘리먼트 단위와 같은 세밀한 단위의 접근제어를 허용하는가의 여부이다. 세가지 비교 대상 모두 지원이 가능하다.

투명성은 접근제어가 이루어지는 것을 사용자가 알 수 없어야 함을 의미하는 것이다. W3C의 Encryption는 자신의 키로 복호화 되지 않는 부분이 존재하는 것을 사용자가 알 수 있기 때문에 투명하지 않다. Pool Encryption과 제안하는 기법은 투명성을 제공한다.

내용기반 접근제어는 특정 부분의 내용에 따라 권한을 다르게 적용하는 것이다. W3C의 암호화는 권한을 기술하고 있지 않으므로 지원이 불가능하다. 하지만 Pool Encryption과 제안하는 기법은 권한을 암호화 이전에 반영하여 대체정보를 생성하므로 내용기반 접근제어가 가능하다.

권한이 존재하는 사용자만 접근을 허용한다는 것은 권한을 가진 사용자만이 해당 부분을 복호화 할 수 있다는 것을 의미한다. 세 비교대상 모두 권한이 존재하는 사용자만이 그 내용에 접근할 수 있다.

엘리먼트 단위의 암호화는 문서의 가장 세밀한 단위인 엘리먼트 단위로 암호화를 수행할 수 있는 지의 여부를 나타낸다. 세 비교대상 모두 엘리먼트 단위의 암호화를 지원한다.

권한 사용자 변동 시 문서 수정은 문서의 권한에 사용자가 추가, 제거 등과 같은 변동사항이 존재할 때 문서를 수정해야 하는지의 여부를 의미한다. W3C Encryption는 추가거나 삭제된 사용자를 위해 슈퍼 암호화를 다시 수행해야 한다. 하지만 Pool Encryption는 각각 기밀노드의 허용 사용자를 유지하고 있는 정보집합에 사용자를 추가하거나 제거하기만 하면 되므로 문서의 수정은 불필요하다. 또한 제안하는 기법 역시 사용자만 해당 역할에서 추가 혹은 삭제 하면 되므로 문서의 수정은 불필요하다.

전자서명의 지원은 암호화된 문서에 전자서명을 지원하므로써 무결성과 부인봉쇄를 보장하기 위한 것이다. W3C Encryption는 전자서명을 지원하고 있다. Pool Encryption과 제안하는 기법은 전자서명의 방법은 기술되어 있지 않으나 적용이 가능하다.

information set preservation은 원래 문서의 주된 정보를 정확히 전달하는지를 의미한다. W3C의 Encryption는 정확한 정보를 전달한다. Pool Encryption는 권한이 없는 엘리먼트의 태그 이름을 제공하지 않는다. 이로 인해 의미의 왜곡 가능성이 존재한다. 제안하는 기법은 권한이 없는 상위 노드의 태그 이름을 제공한다. 그러므로 의미의 왜곡은 존재하지 않게 되지만 권한이 없는 사용자가 태그의 이름을 알 수 있는 문제점을 가지게 된다.

package encryption은 여러 개의 엘리먼트를 묶어서 암호화 할 수 있는 지를 의미한다. Pool Encryption는 노드단위의 암호화만 지원하므로 불가능하지만 W3C Encryption과 제안하는 기법은

엘리먼트 집합 단위의 암호화를 지원한다.

접근권한의 결정은 문서를 암호화할 때 보안관리자의 개입 여부이다. W3C Encryption는 문서를 암호화할 때마다 보안관리자가 판단을 하여 수동적으로 암호화를 수행하지만 Pool Encryption과 제안하는 기법은 보안관리자가 기술한 권한에 따라 자동적으로 암호화가 수행된다.

표 1. 제안하는 기법과 기존 기법과의 비교

비교기준	W3C Encryption	Pool Encryption	제안하는 기법
다양한 조직 단위 권한 지원	개인단위 권한	개인단위권한	주체(역할)단위 권한
Fine-grained AC	지원	지원	지원
투명성	불가능 (복호 불가능 부분 존재)	가능 (복호 가능 부분만 존재)	가능 (복호 가능 부분만 존재)
내용기반 접근제어	불가능	가능	가능
권한이 있는 사용자만 접근	가능	가능	가능
엘리먼트단위 암호화	가능	가능	가능
권한 사용자 변동시 문서수정	필요(암호화 수행)	불필요(기밀노드의 권한 사용자만 추가)	불필요 (사용자만 역할에 할당)
전자서명지원	지원	지원가능	지원가능
Information set preservation	가능	권한이 없는 태그 이름을 숨김으로써 의미 변질우려존재	의미 변질우려는 없지만 권한없는 태그 이름을 보여줌
Package encryption	지원	노드단위 암호화만 지원	지원
접근권한 결정	보안관리자가 수동적으로 수행	자동	자동
키의 분배	중첩 암호화 수	기밀노드 수	포함 역할 수 + α ≤ 기밀노드수
전처리과정(추가정보유지)	불필요	각 기밀노드마다 권한 사용자 정보 유지	역할별 대체정보유지
문서의 재조합	불필요	필요	필요

키의 분배, 암호화 수행과정, 문서의 재조합은 성능상의 비교 분석에서 자세히 살펴본다. 또한 제안하는 기법은 따라 기존의 접근제어[5]의 권한 평가과정에서 수행되었던 레이블링과 반복적은 DOM 트리의 방문을 사용자가 요청을 한 후에 수행하는 것이 아니라 전처리 과정을 통해 문서의 생성시에 한번만 수행하게 된다. 전처리 과정으로 대체정보를 생성한 뒤 사용자의 요청이 들어오면 권한의 평가과정 없이 대체정보의 확인만으로 사용자에게 권한이 존재하는 부분만을 전달하게 된다. 또한, 전처리 과정에 암호화를 한번 수행하고 암호화된 문서의 부분을 재사용함으로써 기존의 접근제어에서는 매번 요청할 때 마다 암호화를 수행해야만 했지만 제안하는 기법은 전처리 과정에서 한번만 수행하면 된다. 하지만 제안하는 기법은 문서의 생성시 권한을 평가하는 전처리 과정을 겪게 되므로 권한이 변경되는 상황이 자주 발생하는 환경에서는 적합하지 않다.

■ 성능상의 비교

본 논문의 성능상의 평가를 위해 기존 접근제어와 암호화를 통합한 기법인 XML Pool Encryption[6]과 예를 통한 수행상의 비교를 수행한다. 비교는 다음과 같은 과정에서 수행한다.

- □ 암호화 처리
- □ 키의 개수에 따른 분배와 관리
- □ 암호화된 문서의 부분의 권한 사용자 정보를 유지하기 위한 추가정보 유지
- □ 암호화된 노드의 분리저장에 따른 위치 복원 처리

예제를 수행하기에 앞서 본 문에 제시한 예제에 대한 수정이 필요하다. 제안하는 기법은 역할을 기반 으로 한을 기술한 것이고 Pool Encryption 는 사용자 관점에서 역할을 기술한 것이기 때문에 각 역할에 대한 사용자의 지정이 필요하다. 또한 본문의 예제는 모든 문서가 기밀로 이루어진 것이어서 사용자는 어떠한 노드도 권한 없이는 접근할 수 없다. 하지만 Pool Encryption는 문서전체 보다는 각 노드단위의 암호화를 수행함으로 본문의 예제는 적합하지 않다. 그러므로 본문의 예제의 권한정책을 일부 수정하여 비교한다.

■ 역할의 사용자 지정

□ staff : 옥기, 경진
□ billing_staff : 진희, 근혜
□ doctor : 지연, 진상
□ head_doctor : 아영

■ 수정된 권한

<staff, Medical.xml:/MedicalRecord/personal info, read, +, R>
<staff, Medical.xml:/MedicalRecord/Medical_characteristic, read, +, R>
<billing_staff, Medical.xml:/MedicalRecord/billing info, read, +, R>
<doctor, Medical.xml:/MedicalRecord/Medical istory/case[./@type='confidential"], read, -, R>
<head_doctor,Medical.xml:/MedicalRecord/Medicalhistory/case[./@type='confidential"], read, +, R>
<head_doctor,Medical.xml:/MedicalRecord/Medicalhistory/case[./@type='sensitive"], read, +, R>
<doctor,Medical.xml:/MedicalRecord/Medical history/case[./@type='sensitive"], read, +, R>

위에 기술된 권한은 모든 노드의 암호화를 피하기 위해 staff에게 허용된 부분은 암호화 하지 않는다. 또한 Pool 암호화에서는 권한의 상속을 허용하지 않기 때문에 본문 예제의 권한기술과는 다르게 상속 없이 모든 권한을 기술한 것이다.

■ 암호화 처리

Pool Encryption는 기밀 노드 각각을 다른 키로 암호화를 수행하고 제안하는 기법은 주체기반으로 키를 생성하고 암호화한다. 예제를 기술된 권한에 따라 기밀노드를 표기하면 그림 12와 같다.

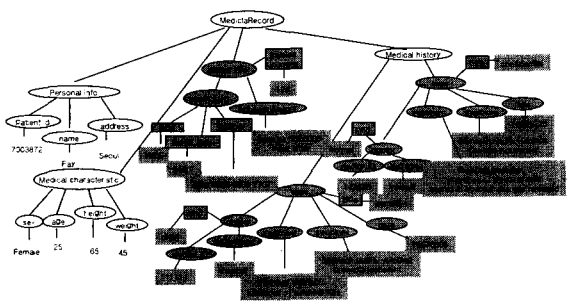


그림 13. 기밀 노드가 표기된 예제

표 2. 제안하는 기법과 기존 기법과의 성능상 비교

	XML Pool Encryption	제안하는 기법
암호화 횟수	25번(노드별)	3번(주체별)
키의 생성 수	25번(노드별)	3번(주체별)
추가정보 유지	7(billing_info)×2(billing_staff 수)+9(confidential)×1(head_doctor)+9(sensitive)×3(doctor,head-doctor)= 50개	1+2+2+1×2(staff)+1×2(billing_staff)+1×2(doctor)+1 × 1(head_doctor)=12개
위치복원정보	전체노드검색 2번 모든트리노드(36) × 정수값 2개	Path 정보 3개

Pool Encryption는 위의 기밀노드를 각각 다른 키로 암호화하게 된다. 기밀노드는 25개가 존재하므로 25번의 암호화가 수행된다. 반면 제안하는 기법은 암호화노드에 권한을 반영한 주체의 역할기반의 그룹을 형성하여 3개의 unit을 위한 3번의 암호화를 수행하면 된다.

■ 키의 개수에 따른 분배와 관리

문서를 암호화할 때 키의 개수는 생성과 분배의 관리적 측면에서 중요하다. 키의 개수가 많아지게 되면 생성에서도 문제가 될 뿐 아니라 사용자에게 알맞은 키를 분배할 때 도 많은 어려움을 갖게 된다. 예제의 경우 기밀노드가 25개 존재하므로 각각의 키로 암호화를 수행하면 25개가 필요하게 된다. 키는 기밀노드의 아이디와 함께 키 풀에 저장되게 된다. 반면 제안하는 기법은 각 주체 역할의 키 즉, doctor, head_doctor, billing_staff의 키로 3개만 존재하면 된다.

■ 암호화된 문서의 부분의 권한 사용자 정보를 유지하기 위한 추가정보 유지

사용자의 요청이 들어오면 사용자에 권한에 알맞은 문서를 제공하기 위해 기밀노드에 어떤 사용자가 허용되는지에 관한 정보를 유지하고 있어야 한다. Pool Encryption는 이를 위해 기밀노드마다 권한 사용자들의 집합을 유지하고 있어야 한다. 예제의 경우 7(billing_info)×2(billing_staff수)+9(confidential)×(head_doctor)+9(sensitive)×(doctor,head-doctor)= 50개의 정보를 유지해야 한다. 반면 제안하는 기법은 같은 권한을 가진 그룹별로 어떤 역할에 허용되는지와 그 역할에 어떤 사용자가 할당되는지의 정보만을 유지하면 된다. 각

역할에 해당하는 사용자가 많고, 기밀노드가 많은 환경에서 사용자를 역할에 할당하고 기밀노드의 서브 트리를 형성해서 하나의 단위로 암호화를 수행하므로서 수행상의 이점을 얻을 수 있다.

■ 암호화된 노드의 분리저장에 따른 위치 복원 처리

Pool Encryption는 문서 전체 트리에 인접리스트모드를 이용해 위치 정보를 저장하게 되고, 노드를 복호화 할 때 다시 한번 트리의 전체 위치정보를 확인하여 기밀노드의 위치를 복원하게 된다. 이를 위해 전체 트리의 검색이 2번 이루어져야 하고 트리 전체 노드마다 위치정보를 위해 두개의 정수값을 유지해야 한다. 반면 제안하는 기법은 주체별 서브트리를 형성하여 그룹으로 만들고 그룹의 path 정보를 유지하고 있다가 복호화할 때 위치 복원에 사용하게 된다.

지금까지 여러 수행관점에서 Pool Encryption와 제안하는 기법을 예를 통해 비교해 보았다. 제안하는 기법은 역할과 그 관계로 인한 권한의 상속과 키의 상속을 허용하고 노드단위가 아닌 주체의 역할 단위의 암호화를 수행하므로서 모든 단계에서의 수행상의 이점을 얻을 수 있었다. 특히, 사용자가 많아지고 사용자 각각이 아닌 역할에 따른 권한이 기술되고, 문서에 기밀정보가 많이 존재하는 환경에서 제안하는 기법이 유용하다.

V. 결 론

본 논문은 접근제어와 암호화를 통합하는데 있어 역할기반의 암호화를 제안하였다. 주체기반의 암호화는 이전의 노드 단위의 암호화로 인해 발생했던 문제점들을 해결하고 있다. XML문서의 권한을 반영하여 같은 주체는 같은 키로 암호화를 수행함으로써 키의 생성과 암호화와 복호화의 횟수를 줄일 수 있는 관리상의 이점이 있다. 또한 암호화를 전처리 과정에 수행하고 대체정보를 생성하여 대체정보의 확인으로 권한의 평가를 대신하게 된다. 이것은 접근제어의 단점인 매번 사용자의 요청 후에 권한을 평가하기 위한 레이블링 과정과 DOM 트리의 반복적인 방문으로 발생했던 시스템의 성능저하를 줄일 수 있다. 하지만 제안하는 기법은 권한이 변경되면 전처리 과정을 다시 수행해야 하므로 권한의 변경이 빈번한 환경에서는 적합하지 않다. 이처럼 본 논문은 기업이나 정부와 같은 복잡한 보안정책을 가지는 조직에서 보안정책과 기밀성을 동시에 보장함으로써 기업의 ERP환경이나 전자상거래에 이용될 수 있다. 또한 제안하는 기법은 읽기 연산만을 지원하고 있기 때문에 쓰기 연산을 수행하고 무결성을 보장할 수 있는 암호화 기법에 관한 연구와 전자서명을 지원하는 방법을 기술하여 인증과정과 부인봉쇄까지 포함한 통합적인 XML문서의 보안 모델을 지원하는 연구가 필요하다.

참 고 문 헌

[1] Takeshi Imamura, Blair Dill away, Ed Simon, "XML Encryption Syntax and Processing", *W3C Recommendation* 10 December 2002, http://www.w3.org/TR/xmlenc-core/

[2] Mark Bartel, John Boyer, Barb Fox, Brian LaMacchia, Ed Simon, "XML-Signature Syntax and Processing", *W3C Recommendation* 12 February 2002, http://www.w3.org/TR/xmldsig-core/

[3] XML Key Management Specification (XKMS) Version 2.0, *W3C Working Draft* 18 April 2003, http://www.w3.org/TR/xkms2/

[4] Simon Godik, Tim Moses, et el, "eXtensible Access Control Markup Language (XACML) Version 1.0", *OASIS Standard*, February 18th, 2003

[5] E.Damiani, S.Vimercati, S.Paraboschi, and P.Samarati, "Securing XML Documents" *In Proceedings of the 2000 International Conference on Extending Database Technology(EDBT2000)*, Konstanz, Germany, March 27-31, 2000.

[6] Christian Geuer-Pollmann, "XML Pool Encryption", *In Workshop On Xml Security Proceedings of the 2002 ACM workshop on XML security*, Nov. 22, 2002

[7] AlphaWorks, "XML Security Suite," http://www.aophaWorks.com/tech/xnl

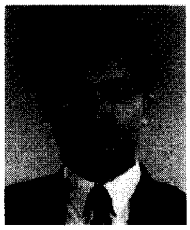
securitysuite (1999).
[8] Chung-Hwan LIM, Seog Park, "Access Control of XML Documents Considering Update Operations", *In Proceedings of the 10th ACM workshop on XML security*, October 31,2003 Fairfax, VA, USA
[9] Hristo Koshutanski, Fabio Massacci, "An access control framework for business processes for web services", *In Proceedings of the 10th ACM workshop on XML security*, October 31, 2003 Fairfax, VA, USA
[10] Jingzhu Wang, Silvia Osborn, "A role-based approach to access control for XML databases", *In Symposium on Access Control Models and Technologies*, June 2,2004 Yorktown Heights, New York, USA
[11] 박 석, 최동희, "XML문서의 접근제어에 관한 연구", *데이터베이스연구회지* 제 19권 제1호, 2003. 3
[12] 김주한, 문기영 , "XML기반 접근제어 기술동향", *한국정보보호학회지*, 2003. 8

[부록] 전처리 과정의 알고리즘

```
Pre-processing algorithm
Input : XML document URI
Output : replacement information, keys,
         encryption element set
/*L is Local, R is recursive*/
1. A:={a∈autorization policy | uri(object(a)) =URI}
2. 문서URI에 해당하는 트리 T, 트리 T의 root를 r이라 하자.
3. for each a do initial_label(n∈object(a))
4. conflict_resolution (r)
5. G is node set group by having same label set
6. for each G do replacement_information(G)

procedure initial_label(n)
/* 노드 n의 레이블을 labeln 이라하고, labeln은 Sn과 Pn으로 구성된다 */
1. Sn := subject(a)
2. Pn := sign(a)
3. if type(a) == R than label(children(n))

procedure conflict_resolution(r)
/*노드 n의 레이블이 2개 이상이면 labeln_1, labeln_2, …, labeln_n으로 구성된다*/
1.if (labeln) > 1 than
      1a.if Pn(labeln_1)==Pn(labeln_2)
            than remove labeln_2
         else remove non_specific labeln
      1b.if (labeln)>1 than remove labeln(Pn==+)

procedure replacement_information(g)
/*그룹 g의 레이블 labelg 이라하고, labelg은 Sg과 Pg으로 구성된다 */
/*그룹 g의 레이블이 2개 이상이면 labelg_1, labelg_2, …, labelg_n으로 구성된다*/
1.if (labelg) > 1 than
1a. if Pg(labelg_1)==Pg(labelg_2) than
      1aa.if Sg(labelg_1)<Sg(labelg_2) than
         create_unit(labelg_2)
            else for each labelg do create_unit(labelg)
   else for each labelg do create_unit(labelg)
else for each labelg do create_unit(labelg)

procedure create_unit(labelg)
/*unit은 subject,path,sign,eid의 어트리뷰트로 구성된다.*/
1. subject(unit) := Sg(labelg)
2. path(unit) := path expression from parent(unit) to start labeled node
3. sign(unit) := Pg(labelg)
4. if sign(unit)==+ than e_id(unit) := Encryption()
```

procedure Encryption()
1. create e_id
2. if (unit) >1 than
2a. generate key k
2b. encrypt with k and store encryption element set
2c. forward <eid,k> to key server
3. else
3a. if not exist subject(unit)'s key than
3aa. generate subject(unit)'s key s
3b. encrypt with s and store encryption element set
3c. forward <eid,s> to key server
4. return(e_id)

〈著者紹介〉



박 석 (Seog Park)
1978년 2월 : 서울대학교 계산통계학과 졸업
1980년 2월 : 한국과학기술원 전산학과 석사
1983년 2월 : 한국과학기술원 전산학과 박사
1983년~현재 : 서강대 컴퓨터학과 교수
1989년~1991년, 2002년~2003년 : University of Virginia 방문교수
1996년 7월~1998년 12월 : 한국정보과학회 데이터베이스연구회 운영위원장
1997년 1월~현재 : 한국정보보호학회이사
2005년 1월~현재 : 한국정보과학회 부회장
〈관심분야〉 XML, RBAC, 데이터베이스 보안, 센서네트워크 데이터 관리, 트랜잭션관리, 웹 서비스

최 동 희 (Dong-Hee Choi)
2000년 8월 : 연세대학교 전산학과 졸업
2004년 2월 : 서강대학교 컴퓨터학과 공학석사
2004년 2월~현재 : 국방과학연구소
〈관심분야〉 XML, RBAC, 데이터베이스 보안, 웹 서비스 보안