다중 역할 계층을 위한 암호학적인 키 할당 기법

배경만*, 반용호, 김종훈
*동아대학교 컴퓨터공학과
e-mail:mang80@donga.ac.kr

Cryptographic Key Assignment Solution For A Multi-Role Hierarchy

Kyoung-Man Bae*, Yong-Ho Ban, Jong-Hoon Kim
*Dept of Computer Science, Dong-A University

요 약

구조적 문서에 대한 접근제어를 위해서 필수적으로 보안 계층에 대한 문제가 고려되어야 한다. 본 논문에서는 사용자가 가지는 보안계층, 즉 사용자 역할 계층에서의 접근제어 문제를 해결하기 위한 효율적인 키 관리 방법을 제안한다. 본 논문에서 제안된 방법은 OWHF(One-way Hash Function)를 이용하여 효과적으로 키를 할당하고 유도한다. 제안된 방법을 역할 계층 트리에서 역할의 추가와 삭제, 역할 계층의 갱신과 같은 동적 접근제어 문제에 적용하고 이를 분석한다.

1. 서론

최근 XML(eXtensible Markup Language)과 같이 구조화된 문서의 사용이 확대되고 있다. XML 문서는 인터넷을 기반으로 하는 환경에서 정보 교환 및 표현을 위해 사용되고 있는데, 인터넷 환경은 외부에서 사용자가 원거리에서 접근하여 일정한 작업을 수행하는 형태로 이루어진다. 즉, 인터넷 환경의 특성상 의도적이던 비의도적이던 구축된 XML 문서는 언제든지 외부에 노출될 수 있음을 의미한다. 이러한 이유 때문에 특정 환경에서 사용되거나 구축된 XML문서에 대한 보안 문제가 계속 이슈화 되고 있는 실정이다. 이러한 문제는 기존 연구에서 다양한 접근 방법들이 제시되었다. 그러나 기존에 제시된 연구들의 대부분은 안전하지 않은 통신상의 전송 문제에 초점을 맞추고 있다. 하지만 XML 문서의 사용의 확대는 하나의 XML문서에 포함된 정보가 그 보안성을 달리하고 있다는 새로운 문제점이 제기되었으며, 이에 대한 새로운 접근방법들이 제시되고 있다.[2][3][4] 즉, 기존에 고려되었던 특정 문서에 대한 전송상의 보안만이 아니라, 접근 제어와 같은 관리적인 측면에서의 보안 문제도 반드시 고려되어야 한다. 예를 들어, 기업 환경에서 사원에 대한 정보나 각 부서별 업무에 대한 정보들은 모든 사람들에게 공통적으로 공개될 수 없는 매우 민감한 정보들이다. 사원의 개인 신상에 관한 정보나 각 부서의 주요 업무에 관한 정보들은 그 정보를 볼 수 있도록 승인된 한정된 인원에게만 공개되어야 하며, 승인된 인원 역시 이들 정보를 이용해 다른 정보를 유추할 수 없도록 하는 새로운 보안정책과 메커니즘이 필요하다. 본 논문에서는 이러한 보안정책과 메커니즘을 구성하기 위해 특정 문서에 대하여 접근 가능한 사용자 역할 그룹을 정의하고, 사용자가 가지는 보안 계층, 즉 사용자 역할 계층에서의 접근제어 문제를 해결하기 위한 효율적인 키 관리 방법을 제안한다. 본 논문에서 제안된 방법은 OWHF (One-way Hash Function)를 이용하여 효과적으로 키를 할당하고 유도한다. 제안된 방법을 역할 계층 트리에서 역할의 추가와 삭제, 역할 계층의 갱신과 같은 동적 접근제어 문제에 적용하고 이를 분석한다. 본 논문의 2절에서는 1절에서 언급된 내용들에 대한 이전의 연구에 대하여 간략하게 언급한다. 논문의 3절에서 역할계층을 위한 키 관리 방법을 제안하고, 제안된

접근방법에 대한 특징과 보안성을 분석한 결과를 제시한다. 마지막으로 4절에서 결론 및 추가적인 연구 방향에 대하여 언급한다.

2. 관련 연구

2.1 계층에서의 접근제어 문제

접근제어 문제를 해결하기 위해서 역할에 대한 고려는 매우 중요하다. 역할은 어떤 데이터에 접근하기 위해 인증되어져야 한다. 이러한 상황을 고려해서 사용자들은 $S=\{S_1,S_2,.....,S_n\}$과 같은 집합으로 나누어지고, 각 집합 S_i는 노드 U_i로 나타낼 때, 각 사용자는 보안 등급(security clereance)에 따라 적당한 보안 계층에 할당된다. 각각의 노드는 기호 ≤에 의해 부분적으로 정렬될 수 있다. 예를 들어, $U_i \leq U_j$의 의미는 S_i의 사용자는 보안 등급(security clereance)이 S_j 보다 낮거나 같다는 의미이다. 즉, S_j는 S_i가 가지는 정보에 접근할 수 있다. 그러나 반대의 경우는 허락되지 않는다. 이 문제를 계층에서의 접근 제어 문제라 부른다.[5][6][8][9]

2.2 RBAC 접근제어

역할 기반 접근 제어 모델(RBAC)에서는 사용자가 어떤 작업을 수행하던지 그 사용자에게 역할을 정의하고 역할 계층은 역할 사이에 반영된 권한의 상속과 책임을 보여준다.[1] 역할 계층은 다음과 같이 정의된다. 만약, $r_i \rightarrow r_j$ 라면 역할 r_i는 r_j 의 모든 권한을 상속받는다. 역할 r_i는 r_j의 직접적인 부모 역할(DPR)이라 부른다. r_j는 역할 r_i의 직접적인 자식 역할(DCR)이라 부른다. 만약 $r_i \rightarrow r_j$ 이고, $r_j \rightarrow r_k$, $r_i \rightarrow r_k$ 이면 역할 r_i는 역할 r_k의 간접적인 부모 역할(IPR)이라 부르고, 역할 r_k는 역할 r_i의 간접적인 자식 역할(ICR)이라 부른다. 그림 1.에서 역할 계층의 한 예를 보여주고 있다. 역할 *R1*는 역할 *R3*와 *R4*의 직접적인 부모이고 역할 *R2*는 역할 *R4*의 직접적인 부모, 그리고 역할 *R3*는 역할 *R5, R6*의 직접적인 부모이다. 비슷하게 역할 *R4*는 역할 *R6*의 직접적인 부모이고 역할 *R2*는 역할 *R7, R8*의 직접적인 부모, 역할 *R6*의 간접적인 부모이다.

2.3 접근제어를 위한 키 할당

RBAC에서 키를 가지는 역할의 각 위치는 관리자에 의해 할당된다. 이러한 환경에서 제기되는 문제는 역할 계층에서 더 높은 레벨에 있는 역할의 사용자가 직적접이거나 간접적인 자식 역할들에게 키를

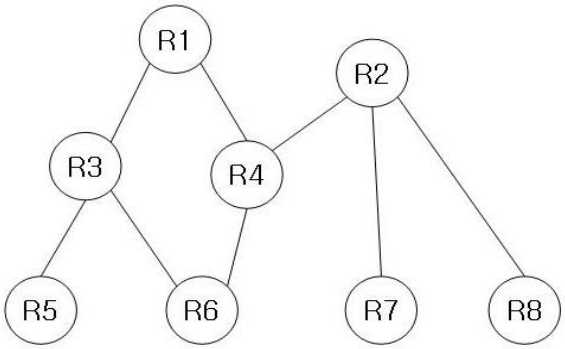

그림 1. 역할 계층의 예

어떻게 파생시킬 것인가 하는 것이다. 가장 간단한 방법은 직적접이거나 간접적인 자식 역할들의 모든 키를 가지는 것이다. 그러나 보안상의 문제 때문에 모든 역할의 키를 관리하는 것은 어렵다. 또한 키를 유지하는데 과도한 자원을 소비하게 된다. Akl 과 Taylor는 암호학적인 접근을 사용하여 부분적으로 정돈된 다중 레벨에서의 키 분산 문제에 대한 해결 방법을 제시하였다.[5] 이 접근법에서 *Central Authority (CA)*는 키의 생성과 키의 분배의 책임을 가진다. 만약 $U_i \leq U_j$ 라면 공개 정수 t_i를 각 노드 U_i 에 t_i/t_j 속성과 함께 할당한다. CA는 키 K_0 와 비밀 소수 p , q를 랜덤하게 선택한다. $M=p*q$ 라는 결과물을 공개하고 $K_j = K_0^{\,ti} \bmod M$은 U_i 인 사용자에게 분배한다. 만약 $U_i \leq U_j$ 의 경우 ti/tj 는 정수이면 U_j 는 $K_j = K_0^{\,ti} = (K_0^{\,ti})^{ti/tj} \bmod M$ 에 의해 K_i를 계산 할 수 있다. 그러나 만약 $U_i \leq U_j$ 이고 ti/tj 가 정수가 아니라면 이 계산은 이루어질 수 없다. 이러한 접근방법의 장점은 키 생성과 키의 유도 방법이 매우 간단하다는 것이다. 그러나 이러한 방식의 약점은 계층에서 security class의 수가 증가할 때 각 security class의 공개 파라미터를 저장하기위한 많은 저장 공간이 필요하다는 점이다.

하지만 정수의 크기는 네트워크 상의 노드의 수와 비례한다. 노드의 수가 아주 커진다면 이 방법은 실용적이지 못하다. 이 논문에서 우리는 역할 계층에서 발생하는 접근제어 문제를 해결 할 암호화 키 관리 방법에 대해 말한다. 우리의 해결 방안은 각 역할의 키는 OWHF(One-way Hash Function)를 이용해서 지역적으로 계산하는 것이다. 이 방법은 키 유도 시 요구되는 공개 파라미터가 줄어드는 장점을 가진다.

3. 역할 계층을 위한 키 관리 방법

3.1 역할을 위한 키 할당과 키 유도

본 논문에서 각각의 역할에 대해 키를 유도하기 위하여 OWHF를 사용한다. 역할 계층이 주어지면 우리는 One-Way Hash함수 $H_j :\{ H_1, H_2,....,H_n \}$를 선택한다. n은 역할 계층 안에서 직접적인 자식 역할의 최대 수이다. Hash 함수는 공개한다. 그림 1.의 역할 계층에서 키의 생성과 유도는 그림 2.에서 보여준다. 만약 역할이 역할 계층에서 직접적인 부모 역할을 가지지 않는 다면 우리는 이 역할을 dead-end 역할이라 부른다. 예를 들어, 그림.1에서 역할 *R1*와 역할 *R2*는 dead-end 역할이다.

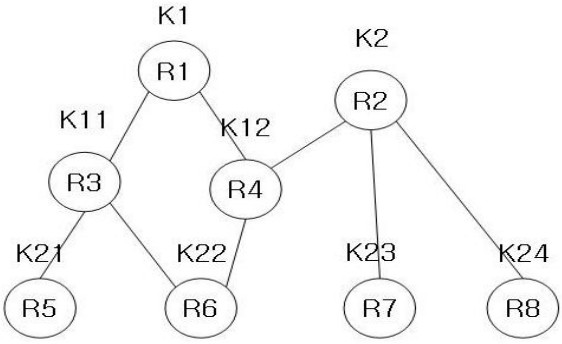

그림 2. 역할 계층을 위한 키 유도

역할 계층에서 역할의 키는 *Central Authority (CA)* 에 의해서 생성된다. 그 과정은 다음과 같다.

(1) CA는 각 dead-end 역할에 임의의 키를 할당한다.

(2) 만약 역할 r_j 가 오직 하나의 직접적인 부모를 가지면 그 부모의 키는 K이다. 그리고 만약 역할 r_j 가 부모의 I번째(왼쪽에서 오른쪽으로) 직접적인 자식이라면 역할 r_j 의 키는 $H_j(K)$ 이다.

(3) 만약 역할 r_j 가 많은 직접적인 부모 역할 ($r_j^1, r_j^2,.....,r_j^m$) 를 가지고 r_j 가 제일 왼쪽의 직접적인 부모인 r_j^1 의 I번째 직접적인 자식이고, 만약 r_j 가 r_j^2의 k번째 직접적인 자식이고, r_j 가 r_j^m 의 n번째 직접적인 자식이면 r_j ($r_j^1, r_j^2,.....,r_j^m$)의 직접적인 부모의 키는 $K_1, K_2,....,K_m$ 이고, r_j 의 키는 $H_j(H_i(K_1), H_j(K_2),.....,H_n(K_m))$, $(1 <= I, k, n <= m)$ 이다. 또, 이 역할은 r_j 는 r_j 의 직접적인 부모와 간접적인 부모들의 키를 H_j 를 이용하여 해시 값을 만들고 그 값들을 공개 값으로 가진다.

그림 2.의 역할 계층에서 각 역할의 직접적인 자식의 최대 수는 3개이다. 따라서 우리는 H_1, H_2, H_3 3개의 Hash 함수를 선택한다. 키 *K1*은 dead-end 역할 *R1*에 할당되고 키 *K2*는 dead-end 역할 *R2*에 할당되었다. 역할 계층에서 다른 역할을 위해 키를 유도 하는 방법은 다음에서 언급한다. 오직 하나의 직접적인 부모를 가진 역할, 역할 *R3* 의 키 $(K11)$는 $H_1(K1)$일 것이다. 역할 *R4*와 같이 더 많은 직접적인 부모를 가진 역할의 키는 *K12*이고, 이것은 $H_2 (H_2 (K1), H_1(K2))$ 일 것이다. 그러나 역할 *R1*와 *R2*에서 역할 *R4*의 키를 유도 하기위해서는 특별한 공개파라미터를 알아야 한다. 역할 *R2*의 사용자는 $H_2 (K1)$ 의 값을 알고 역할 *R1*의 사용자는 $H_1(K2)$ 의 값을 알아야 한다. 우리의 방법은 몇 개의 공개 파라미터를 사용하지만 Akl 과 Taylor가 제안한 방법 보다는 많지 않다. 하나 이상의 직접적인 부모를 가지는 역할들만이 공개 파라미터를 가지기 때문이다.

3.2 동적 접근 제어

이 절에서는 역할의 추가, 삭제, 관계의 변경과 같은 동적 접근 제어 문제에 대해 설명한다.

▪ 역할의 추가

만약 추가된 역할 *R*이 daed-end 역할이면 CA는 역할 *R*의 새로운 키와 *R*의 직접적인 자식과 간접적인 자식들의 키를 새로 생성한다. 만약 추가된 역할 R이 daed-end 역할이 아니라면 CA는 역할 *R*의 키는 직접적인 부모로부터 유도 된다. 역시 *R*의 직접적인 자식과 간접적인 자식들은 키를 새로 생성한다.

▪ 역할의 삭제

만약 삭제된 역할 *R*이 daed-end 역할이고 *R*이 삭제되고 새로운 daed-end 역할이 존재하면 CA는 새로운 daed-end 역할의 키를 변경하지 않는다. 만약 그렇지 않으면 R의 직접적인 자식과 간접적인 자식들은 키를 새로 생성한다. 예를 들어 그림. 3에서 우리는 역할 *R1*를 삭제했다고 가정했을 때 역할 *R3*는 역할 *R1*이 삭제된 후에 새로운 daed-end 역할이 된다. CA는 역할 *R3*의 키를 바꾸지 않는다. CA는 오직 역할 *R4*의 직접적인 자식과 간접적인 자식들의 키를 새로 생성한다.

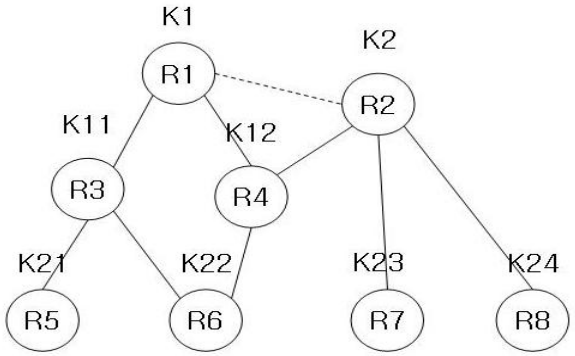

그림 3. 역할 계층에서의 동적 접근 제어

▪ 역할 관계의 변경

역할 *R*을 역할 *S*의 직접적인 부모 역할이고 역할 *R*과 역할 *S* 사이의 관계를 삭제했다고 가정한다. 만약 역할 *S*가 역할 *R*이 삭제되고 난 후 daed-end 역할이 된다면 CA는 *S*의 키를 바꾸지 않는다. 만약 그렇지 않으면 *S*의 키와 *S*의 직접적인 자식과 간접적인 자실들의 키를 새로 생성한다. 역할 *S*와 역할 *R*이 서로 관계(직접적이거나 간접적인 부모나 자식 역할)를 가지로 있지 않다고 가정 했을 때 역할 *S*와 역할 *R*사이에 새로운 관계가 추가되고 *S*는 *R*의 직접적인 부모가 된다면 그 후 *R*과 그 자식들의 키는 새로 생성된다. 예를 들어 만약 그림.3에서 역할 *R1*와 역할 *R2*사이에 관계가 추가 된다면 역학 *R2, R4, R6, R7, R8*의 키들은 새로 생성된다.

3.3 제안된 키 관리 방법의 보안성 분석

먼저, 제안된 방법은 OWHF이라는 안전한 암호 시스템을 기반으로 하였기 때문에 보안상 안전성이 보장된다. 또한, 역할 기반 접근 제어 시스템에서는 사용자가 어떤 작업을 수행하던지 그 사용자에게 역할을 부여한다. 역할의 구성원은 역할에 대응되는 키를 얻는다. 만약 역할의 구성원으로서 해당 사용자가 삭제된다면 이 역할을 위한 키 역시 제거되어 더 이상 해당 적업을 처리하지 못하게 된다. 또한 역할 *R*의 하나의 직접적인 부모를 가지는 역할의 S의 사용자는 몇 개의 공개 파라미터를 가질 수 있다 그러나 사용자는 역할 *R*의 다른 직접적인 부모의 키는 얻지 못한다. 그 사용자의 정보는 오직 키의 해시 값이다. 예를 들어, 그림2에서 역할 *R1*의 사용자는 특별한 파라미터 정보인 *H1(K2)* 를 역할 *R2*로부터 가지고 온다. 그러나 그 사용자는 역할 *R2*의 키 *K2*는 상속 받지 못한다. hash 함수는 반대로 계산이 불가능하기 때문이다.

4. 결론

우리는 본 논문에서 역할 계층에서의 키 관리 방법을 제안하였다. 본 논문에서 제안한 방법은 동적 접근 제어의 경우 역할 계층 구조가 변할 때 역할의 일부분만 변경하게 하고 OWHF를 사용함으로써 작은 저장 공간만 필요하게 되었다. 본 논문에서 제안한 방법은 분산 시스템이나 계층적인 구조를 가지는 곳에 적용할 수 있다.

본 논문에서 제안한 방법은 위에서 언급한 장점들이 있지만 보다 복잡한 계층 구조에서의 키 할당 시 생기는 문제점들에 대한 추가적인 연구가 필요하다.

참고문헌

[1] R. Sandhu, E,J.Coyne, H.L.Feinstein, “Roles based access control model", IEEE computer, 29(2), 33-47, 1996

[2] Specifying and Enforcing Access Control Policies for XML Document Sourecs, Elisa Bertino, Silvana Castano, 2000

[3] E. Damiani, S. D. C di Vimercati, S. Paraboschi, P. Samarati, "Controlling access to xml documents" IEEE Internet Computing 5(6):18-28, November 2001.

[4] 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.05 No.02 169-202, 2002.

[5] S.G. Akl, P.D. Taylor, Cryptographic solution to a problem of access control in a hierarchy, ACM Trans. Comput. Syst. 1 (1983) 239 - 248.

[6] C.H. Lin, Dynamic key management schemes for access control in a hierarchy, Comput. Commun. 20 (1997) 1381 - 1385.

[7] R.S. Sandhu, Cryptographic implementation of a tree hierarchy for access control, Inform. Process. Lett. 27 (1988) 95 - 98.

[8] Cungang Yang, celia Li and Richard Cheung, Cryptographic Key Management Solution In a Role Hierarchy. CCECE2004-CCGEI2004, Niagara Falls, May/mai 2004.

[9] Tzer-Shyong Chen *, Jen-Yan Huang, A novel key management scheme for dynamic access control in a user hierarchy. Applied Mathematics and Computation 162 (2005) 339 - 351