정보학연구 제4권 제4호 2001년 12월(pp.15-24)
The Journal of Information Technology, Vol.4, No.4

확장된 역할기반 접근제어 모델에서의 사용자 수준의 위임기법

박종화*

요 약

현재의 역할기반 시스템에서 보안 관리자가 사용자에게 역할을 지정하는 일을 하고있다. 이와 같이 보안 관리자에 의한 역할 지정은 분산 환경 하에서 관리의 어려움을 더욱 증대시킬 수 있다. 역할 기반 위임은 사용자에게서 사용자에게로 역할의 권한을 위임하는 것에 의해 분산 환경 하에서 역할 기반 접근 제어를 적용하기 위한 수단을 제공한다. 역할 기반 위임의 기본 개념은 사용자가 어떤 기능을 수행할 수 있는 자신의 역할의 권한을 다른 사용자에게 위임하는 것을 의미한다. 이 논문은 하나의 역할기반 위임 모델을 제시한다. 이 위임 모델은 확장된 역할기반 접근통제(ERBAC : Extended RBAC) 모델에서 사용자 수준의 위임에 대한 구현을 반영한다. 확장된 역할기반 접근제어 모델은 기존의 RBAC모델에 추가로 주체 및 객체를 고려한 모델이다.

Ⅰ. 서론

역할기반 접근제어(Role-Based Access Control)의 개념은 다중 사용자와 다중 응용을 위한 온라인 시스템 상에서 1970년대에 시작되었다. RBAC의 기본 개념은 역할과 권한이 지정되고, 사용자는 적절한 역할에 지정된다는 것이다. 이것은 권한의 관리를 쉽게 해준다. 역할은 조직 내에서의 다양한 작업 함수에 의해 생성되고, 사용자는 그들의 자격과 책임에 따라 역할에 지정된다. 사용자는 하나의 역할에서 다른 역할로 재 지정될 수도 있고, 새로운 응용에서 새로운 권한을 부여받을 수도 있다. 또 필요에 따라 권한이 역할로부터 회수될 수도 있다.

현재의 역할기반 접근제어 시스템에서는 보안 관리자가 사용자에게 역할을 할당하는 일을 하고 있다. 보안 관리자에 의한 사용자의 역할지정은 넓은 범위와 고도로 분산된 환경 하에서 보안관리자의 계속적인 참여로 인해 관리에 대한 어려움을 증가시킬 수 있다. 최근 역할기반 위임(role-based delegation)에 대한 연구가 활발하게 진행되고 있는데, 이는 위임이 행정적인 일을 분산할 수 있으므로 분산 시스템의 확장성을 향상시킬 수 있기 때문이다. 따라서 역할기반 위임은 사용자에게 권한을 위임하는 것에 의해 분산 환경에 RBAC을 적용하기 위한 수단을 제공한다.

위임은 분산 시스템 환경에서 보안을 위한 중요한 요소이다. 간단한 예로, Alice가 자신의 역할을 Bob에게 위임한다. Bob의 요청에 의해, 요청된 서비스가 이미 Alice에 부여되었다면 그 서비스는 Bob에게 허가될 것이다. 자연스럽게, 접근 결정을 위해서 이 위임의 개념을 고려하는 것이 필요하다. 또 하나의 연관된 쟁점은 위임된 역할을 철회하는 것이고, 위임과 철회에 관

* 세명대학교 소프트웨어학과 교수

한 보안정책을 어떻게 규정하고, 어떻게 적용하느냐 하는 것이다. 예를 들어, Alice가 위임된 역할로부터 Bob을 철회하고자 할 때, 하나의 철회 기법이 제공되어야 하고 그리고 보안 정책이 이와 같은 행위를 규정해야만 한다. 이 논문은 확장된 역할기반 접근통제(ERBAC)모델 하에서 하나의 역할기반 위임 모델을 제시하고, 정책을 어떻게 적용할 것인가에 대해서도 논의한다. 본 논문의 나머지는 다음과 같다.

2장에서 본 논문의 동기에 대해서 언급되고, 3장에서는 ERBAC모델[1]과 사용자 수준에서 이루어지는 위임모델[2]의 기본 구성 그리고 위임을 위한 사용자의 범위 속성이 다루어지며, 4장에서는 위임과 철회 모델이 제시된다. 마지막으로 5장에서 결론을 밝힌다.

II. 동기

위임에 대한 요구는 한 사용자가 자원의 접근에 대해 다른 사용자의 입장에서 행동할 필요가 있을 때 발생한다. 다시 말해서 위임이란 한 시스템 내에서 역할의 담당자가 어떤 기능을 수행할 수 있는 자신의 권한을 다른 사람에게 위임하는 것으로 정의 할 수 있다. 많은 종류의 위임이 제안되었는데, 대부분의 위임의 형태는 사용자 대 기계, 사용자 대 사용자, 그리고 기계 대 기계의 위임이다[3, 4, 5, 6].

위임은 두 가지 방식[7]으로 정리되는 데, 관리자 수준의 위임과 사용자 수준의 위임이다. 관리자 수준의 위임에서는 관리 적 하부구조가 위임을 중재한다 즉 보안 관리자가 모든 위임을 중재한다. 또한 사용자 수준의 위임에서는 사용자 시스템이 사용자의 통제하에서 자원에 대한 위임을 중재한다. 두 경우에서 개인적인 사용자의 권한 남용을 막기 위해 사전에 정의된 위임 정책 부과하는 것이 필요하다.

Barka와 Sandhu는 위임에 관계된 특성들을 영속성, 무결성, 그리고 위임의 수준으로 구분한다[5]. 영속성은 시간에 기초를 둔 위임의 형태에 관련된다. 무결성은 역할에 할당된 허가가 어떻게 완전하게 위임되는가에 관련된다. 또한 위임의 수준은 각각의 위임이 그 이상 위임될 수 있는지, 또는 몇 번 위임될 수 있는지를 정의한다.

이 논문은 사용자간의 위임 즉 한 사용자가 그의 역할을 다른 사용자에게 위임하는 사용자 수준의 위임을 위하여, ERBAC[1]의 FRAMEWORK에 기초로 역할 계층을 통한 다단계 위임을 포함하는 사용자 수준의 위임이 다루어지고, 이 위임모델를 위해서 RDM2000[2]이 사용되었다. 그리고 역할이 가진 속성과 사용자의 속성을 비교하고, 역할이 가진 제약을 적용함으로써, 위임역할을 지정할 수 있는 대상이 되는 사용자를 제한하여 비밀성(정보유출)과 무결성(의무분리)을 유지한다.

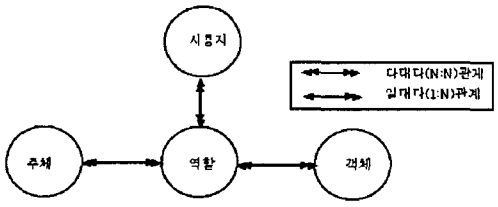
III. 역할 기반 위임과 회수에 위한 FRAMEWORK

이 장에서는 위임 및 회수 모델이 제안되기에 앞서 이 위임 및 회수 모델의 기초가 되는 FRAMEWORK 즉 확장된 역할기반 접근통제 모델(ERBAC : Extended RBAC)과 하나의 위임 모델이 소개된다. 이 ERBAC모델은 Ravi S. Sandhu가 제안한 RBAC모델에 주체 및 객체에

대하여 추가로 고려한 모델이다.

3.1 확장된 역할기반 접근통제 모델(ERBAC)

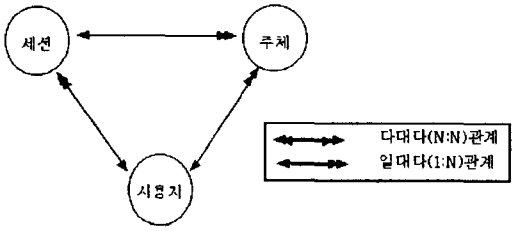
확장된 역할기반 접근통제 모델은 사용자에 대해서만 고려한 RBAC모델에 추가로 주체와 객체에 대해 고려한 모델이다. (그림 1) 및 (그림 2) 는 사용자, 주체 및 객체 와 역할간에 존재하는 관계성(relationship)을 분석 제시하고 있다. (그림 1) 은 사용자와 역할, 주체와 역할, 객체와 역할간의 관계성을 나타낸다. 한 사용자는 여러 개의 역할을 할당받을 수 있고



(그림 1) 사용자, 주체 및 객체와 역할간의 관계

하나의 역할에 여러 명의 사용자가 할당되어 있을 수 있으므로 사용자와 역할간의 관계는 다대다(N : N) 관계가 성립한다. 또한 어느 한 주체(또는 객체)는 여러 개의 역할을 할당받을 수 있고 하나의 동일한 역할에 대해서 다수의 주체(또는 객체)가 할당되어 있을 수 있으므로 주체(또는 객체)와 역할간의 관계는 다대다(N : N) 관계가 성립한다.

(그림 2)는 세션, 주체 및 사용자간의 관계성을 나타낸 것이다. 한 사용자는 다수개의 세션을 유지할 수 있고 한 세션 동안에는 그 세션에 해당되는 단일의 사용자가 연관되어 있으므로 사용자와 세션간에는 일대다(1 : N)의 관계가 성립한다. 또한, 한 사용자는 세션을 유지하는 동안, 다수의 주체를 생성하여 원하는 작업을 실행하게 되며 어느 한 주체는 반드시 단일 사용자에게 연관되어 있으므로 사용자와 주체간에는 일대다(1 : N)의 관계가 성립한다. 따라서 어느 한 세션은 그 세션에 해당하는 주체가 생성하여 실행시킨 다수의 주체들과 연관되므로 세션과 주체간에 일대다(1 : N)의 관계가 성립한다.



(그림 2) 세션, 주체 및 사용자간의 관계

[정의 1] 다음은 ERBAC 모델의 구성 요소이다.

- P, R, U, S, O, 그리고 C는 허가, 역할, 사용자, 주체, 객체 그리고 세션(session)의 집합 이다.
- $PA \subseteq P \times R$ (허가와 역할의 연관관계 집합)
- $UA \subseteq U \times R$ (사용자와 역할의 연관관계 집합)
- $SA \subseteq S \times R$ (주체와 역할의 연관관계 집합)
- $OA \subseteq O \times R$ (객체와 역할의 연관관계 집합)
- $RH \subseteq R \times R$ (역할계층을 나타내는 R에 대한 부분순서의 집합)
- $f_u : C \rightarrow U$ (RBAC 모델에서 함수 user와 동일) $u_i = f_u(c_i)$

- f_s : (C, U) → S (세션 c_j와 세션 c_j에 연관된 특정 사용자 u_i를 주체들의 집합인 $f_s(c_j, u_i)$에게 사상시키는 함수)
  $S_{ji} = f_s(c_j, f_u(c_j)) = f_s(c_j, u_i)$ (임의의 세션 c_j 및 사용자 u_i와 연관된 주체의 집합)
  $S_{ji} = \{s_l, s_m, ..., s_n\}$ (어느 한 사용자 u_i가 자신에게 속한 임의의 어느 한 세션 c_j 동안 원하는 작업을 수행하기 위하여 실행한 주체들은 $s_l, s_m, ..., s_n$이며 세션 c_j가 가질 수 있는 역할은 사용자 u_i에게 부여된 역할과 이 사용자 u_i가 생성한 주체들 즉, $s_l, s_m, ..., s_n$에게 부여된 역할에 의해서 결정된다. (그림 3)은 세션, 주체, 사용자 간의 연관관계를 나타낸다.)
- f_r : (C, U) → 2^R (세션 c_j와 세션 c_j에게 연관된 특정의 사용자 u_i를 역할의 집합인 $f_r(c_j, u_i)$에 사상시키는 함수.)
  $R_{ji} = f_r(c_j, u_i) \subseteq \cup_{s \in S_{ji}}\{r|(s,r) \in SA\}$
  $= \cup_{s \in f_s(c_j, f_u(c_j))}\{r|(s,r) \in SA\}$
  $= \cup_{s \in f_s(c_j, u_i)}\{r|(s,r) \in SA\}$
  (함수 f_u 및 f_s를 이용한 임의의 세션 c_j 및 사용자 u_i와 연관된 역할의 집합)
- f_p : (C, U) → 2^p (세션 c_j 및 사용자 u_i와 연관된 허가의 집합을 결정하는 함수)
  $P_{ji} = \cup_{r \in f_r(c_j, u_i)}\{p|(p,r) \in PA\}$ (함수 f_r을 이용한 임의의 세션 c_j 및 사용자 u_i와 연관된 허가의 집합)

[정의 2] 다음은 주체와 사용자 그리고 역할에 연관된 함수들을 나타낸다.

- SU(s) : S → U (주체와 관련된 사용자를 나타내는 함수)
- SR(s) : S → 2^R (주체가 활성화한 역할의 집합을 나타내는 함수)
- RU(r) : R → 2^u (역할 r에 지정된 사용자의 집합)
- 주체는 관련된 사용자에게 허가된 역할만을 활성화할 수 있다.
  $SU(s) \in RU(r)$

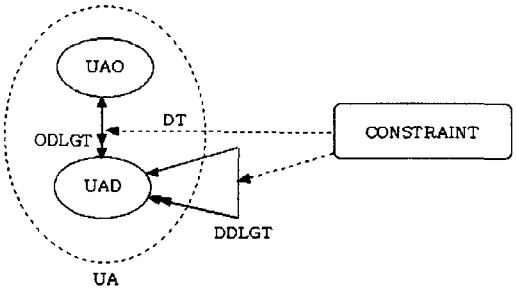
3.2 위임모델의 구성

소개되는 위임모델은 역할 계층에 기초한 사용자 수준의 위임에 초점을 맞추고 있다. 한 사용자로부터 다른 사용자로의 위임 즉 사용자 수준의 위임은 실질적으로 위임할 역할을 한 사용자에게 할당하는 것이다. 따라서 위임할 사용자는 사용자-역할간의 할당을 위한 관계가 필요하다. 다음은 이 위임모델에서 사용되는 구성요소와 시스템 함수들을 소개한다.

[정의 3] 다음은 위임모델에서 사용되는 구성요소들이다.

- UA = UAO ∪ UAD
- UAO ⊆ U × R (original 사용자와 역할간의 다대다 할당관계)
- UAD ⊆ U × R (위임된 사용자와 역할간의 다대다 할당관계)
- Users : R → 2^u (각 역할을 사용자 집합에 사상하는 UA로부터 유도된 함수)
- Users(r) = Users_O(r) ∪ Users_D(r)
- Users_O(r) = {u|(∃r ° ≥r′)(u, r °)∈UAO}
- Users_D(r) = {u|(∃r ° ≥r′)(u, r °)∈UAD}
- DLGT ⊆ UA × UA = U × R × U × R (다대일의 위임관계)
- ODLGT ⊆ UAO × UAD (최초 사용자의 위임관계)
- DDLGT ⊆ UAD × UAD (위임된 사용자의 위임관계)
- DLGT = ODLGT ∪ DDLGT

- DT ⊆ UA × UA (위임 트리)
- Prior : U∪R→U×R (각 UA를 다른UA에 사상하는 함수)
  Prior(u, r) = {(u′, r′)|(u, r) ∈ UAD, (u′, r′, u, r) ∈ DLGT}
  Prior(u, r) = { ∅| (u, r) ∈ UAO }
- 위임 path는 사용자 역할 할당의 순서 집합이다.
  Path(u_0,r_0) = {(u_0,r_0),(u_1,r_1), ..., (u_i,r_i), ...,(u_n,r_n)|(u_i,r_i)=Prior(u_{i-1}, r_{i-1})∈UA, i>0 일 때} Path(u, r) = { ∅| (u, r) ∈ UAO }
- Depth : U∪R→N (위임 depth는 위임 path 안에 원소의 수 minus 일이다.)
  Depth(u, r) = {n|n=|Path(u, r)|, (u, r) ∈ UAD} Depth(u, r) = {0|(u, r) ∈ UAO}



(그림 3) 위임 관계

사용자 수준의 위임은 4개의 요소로 구성된다. 즉, 위임하는 사용자, 위임하는 역할, 위임받는 사용자, 그리고 위임받는 역할 등이다. 예를 들어, (Gail, PL2, Dongwa, QE2)는 역할 PL2를 담당하는 Gail이 Dongwa에게 역할 QE2를 위임함을 의미한다. 위임관계는 다단계 위임 안에서 그 이상으로 나누어지는데, 그것은 최초 사용자 위임(original user delegation : ODLGT)과 위임된 사용자 위임(delegated user delegation : DDLGT)이다. (그림 3)은 이 위임관계를 나타낸다. 이 위임관계에 기초한 함수의 집합에는 함수 Prior, Path, 그리고 Depth 등이 있다. 예를 들면 함수 Prior는 하나의 UA(u1, r1)를 다른 UA(u2, r2) 또는 ∅에 사상한다. 따라서 (u2, r2, u1, r1) 또는 (u1, r1) ∈ UAO 이 존재하게 된다. 함수 Path는 하나의 UA를 위임 path에 사상하고, 함수 Depth는 위임 path의 깊이(depth)를 나타낸다. 하나의 위임 path는 사용자 역할 할당관계의 순서 집합이다. 이 위임 path는 다단계 위임이 적용될 때만 사용된다. 위임 트리(tree)는 사용자-역할 할당/위임의 계층구조이다. 각각의 노드(node)는 사용자-역할 할당을 나타내고, 간선(edge)은 위임관계를 나타낸다. 트리 안에서 사용자-역할 할당의 각 층은 위임 깊이(depth)로서 나타난다.

3.3 사용자 수준의 위임에서 사용자의 범위 속성[8]

사용자 수준의 위임은 현실세계에서 위임을 잘 반영하는 새로운 기법이다. 그러나 사용자 수준의 위임은 보안관리자에 의해 관리될 때보다 많은 보안의 위협을 가진다. 즉, 사용자 수준에서 일어나는 이러한 위임의 전파(propagation)는 허가되지 않은 사용자에게 정보를 유출할 수 있는 문제점이 있다. 따라서 위임할 수 있는 대상에 대한 제한이 필요하고, 이러한 대상을 제한하기 위해서는 기준이 필요하다. 그 기준은 다음과 같이 정리된다.

- 의무분리 정책 : 의무분리 관계에 놓여있는 역할을 위임할 경우에 위임하는 대상이 이미 의무분리 관계에 놓여 있는 다른 역할과 직접 또는 역할 계층을 통해 간접적으

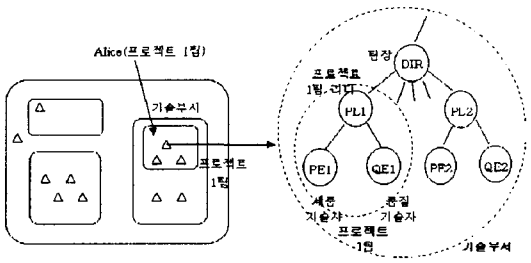
로 지정되어 있다면 위임의 대상으로 적합하지 않다.

- 역할에 지정될 수 있는 사용자 수 : RBAC 허가정책에 부여된 제약의 하나로 조직내에 어떤 역할은 일정 기간동안에 일정 수의 멤버를 가져야만 한다. 예를 들면 부서의 장을 나타내는 역할의 경우, 많은 사용자들이 이 역할에 지정될 수는 있지만, 주어진 한 시점에서 부서의 장은 단 한 명이어야 한다. 위임역할도 기존의 역할과 마찬가지로 이러한 속성을 유지하여야 한다.
- 도메인 내의 사용자 : 역할의 위임은 그 역할을 소유하고 있는 사용자와 관련된 도메인 내에서 이루어지는 것이 일반적이다. 예를 들면, 프로젝트1팀의 한 사용자가 자신의 역할에 대한 권한을 프로젝트2팀의 어떤 사용자에 위임해서는 안된다. 프로젝트의 성격이 중요한 보안을 요구하는 경우에는 더더욱 그와 같은 위임은 일어나서는 안된다. 따라서 같은 도메인, 같은 역할 계층상에서의 위임이 일어나는 것이 바람직하다.

이러한 대상의 제한을 위하여 기존의 역할이 갖는 제약 즉 의무분리 정책과 역할에 지정되는 사용자의 수를 위임역할이 상속하도록 하는 것이다. 다음으로 세 번째 기준인 사용자의 도메인 내에서의 상속은 또 다른 개념을 필요로 한다. 역할은 그것이 정의될 때, 조직의 구성도와 같은 역할 계층에 의하여 역할이 사용되어질 범위가 결정된다. 역할의 범위 속성과 비슷하게 사용자도 처음 등록이 되면 사용자의 범위가 정해진다. 각각의 범위는 역할계층과 유사한 구조를 갖는다. (그림 4)는 이러한 사용자 범위 속성과 역할계층과의 관계를 보여준다.

[그림 4]는 역할계층에서 기술부서(engineering department)를 나타낸다. 기술부서 내에 2개의 프로젝트팀이 운영되고 있다. 각 프로젝트 팀 내에서의 역할 구성과 팀과 부서간의 역할관계는 역할계층을 통하여 나타낸다. 사용자 범위속성의 경우 이와 유사한 의미를 갖는다.

처음에 Alice라는 사용자가 등록되면 이미 정의된 사용자 도메인 상에서 하나의 속성을 부여받는다. (그림 4)의 경우 Alice는 기술부서 내의 프로젝트 팀1의 속성을 부여받았다. 이 경우 Alice는 프로젝트 팀1에 구성되어 있는 역할을 지정 받을 수 있다. 사용자 범위 속성도 역할계층과 같이 계층구조를 갖는다. (그림 4)에서 기술부서의 사용자 속성을 가진 사용자는 하위의 프로젝트 팀1과 프로젝트 팀2의 속성을 모두 포함하기 때문에 두 조직내의 역할에 부여될 수 있다.



(그림 4) 사용자범위 속성과 역할 계층과의 관계

Ⅳ. 위임과 철회

4.1 위임

사용자에게 위임 역할을 위임할 경우에 무결성, 비밀성과 같은 보안 문제를 위하여, 위임 대상자를 제한하는 사용자 범위 속성과 같은 사전

조건(prerequisite condition)을 부과하는 것이 필요하다.

[정의 4] 사전조건(prerequisite condition) CR은 블리언 표현(boolean expression)으로, cr 의 항에 연산자 "&"(and)와 "|"(or)를 사용하는 표현식이다. cr은 x(membership)가 될 수 도 있고, 아니면 -x(non-membership)가 될 수 있다.

[정의 5] 사전조건인 사용자 범위 속성은 다음과 같이 정의된다.

- AS : 범위 속성의 집합
  User_scope_attribute, Role_scope_attribute ∈ AS
- User_scope_attribute = user_scope(u)
  Role_scope_attribute = role_scope(r)
- 다음을 만족할 경우에만 사용자를 역할에 지정할 수 있다.
  u∈RU(r) ⇒ user_scope(u) ⊇ role_scope(r)
- 위임역할(Drole)의 역할범위 속성은 위임할 역할의 범위 속성을 상속받는다.
  role_scope(Drole) = role_scope(r)
- 사용자를 위임역할에 지정할 경우에 다음을 만족해야 한다.
  u∈RU(Drole) ⇒ user_scope(u) ⊇ role_scope(Drole)
- 위임역할(Drole)에 대한 정적 의무분리 관계
  u∈RU(Drole) ∧ u∈RU(j) ⇒ (Drole, j) ∉SSD
- 위임역할(Drole)에 대한 동적 의무분리 관계
  Drole∈SR(s) ∧ j∈SR(t) ∧ (Drole, j) ∈ DSD ⇒SU(s) ≠ SU(t)

[정의 6] 다음의 관계는 사용자 수준의 위임을 나타낸다.

- can_delegate ⊆ R×CR×N
  여기서 R, CR, N은 각각 역할의 집합, 사전조건, 그리고 최대 위임 depth를 나타내다.(r, cr, n) ∈ can_delegate는 역할 r 또는 역할 r의 상위역할(a role senior to role r)에 지정된 사용자는 역할 r 또는 역할 r의 하위역할(a role junior to role r)을 현재 다른역할 t에 지정되어 있으면서 사전조건 cr을 만족하는 다른 사용자에게 최대 위임 depth가 n을 초과하지 않는 범위에서 위임할 수 있음을 의미한다.

4.2 철회

철회는 위임과 함께 동반되어야 하는 중요한 프로세스이다. 위임을 철회하는 방법에는 2가지가 존재하는데, 그것은 위임 기간을 제한하는 강제적인 철회 방법과 사용자에게 철회를 허용하는 방법이다. 이 논문에서는 위임기간을 제한하는 철회 방법은 논의하지 않고, 사용자에 의한 철회만 다루기로 한다.

사용자에 의한 철회는 2가지 형태로 분류할 수 있다. 즉 수여-의존적인(grant-dependent) 철회와 수여-독립적인(grant-independent)철회가 그것이다. 수여-의존적인 철회는 위임 path에서 위임 역할에 지정된 사용자의 바로 이전 사용자가 그 위임 역할을 회수하는 것을 의미한다. 그리고 수여-독립적인 철회에서는 오직 위임 역할의 최초 사용자가 그 역할에 지정된 사용자로부터 그 위임역할을 회수한다.

[정의 7] 위임철회에 대하여 다음과 같이 정의한다.

- can_revokeGI ⊆ R
- can_revokeGD⊆ R

(b)∈can_revokeGI는 현재 위임역할 b에 지정된 사용자는 역할 b의 최초 사용자에 의해서만 위임역할 b로부터 철회될 수 있음을 의미한다. 또 (b)∈can_revokeGD는 위임 path 상에서 위임역할 b에 지정된 사용자 이전의 모든 사용자는 위임역할 b로부터 b에 지정된 사용자를 철회할 수 있다.

V. 결론

본 논문에서 제안한 위임모델은 현실세계에서 위임이 사용자에게서 사용자에게로 일어난다는 점에 중점을 두고, ERBAC 상에서 이러한 사용자 수준의 위임을 구현할 수 있는 방법을 제시한다. 이 모델은 또한 역할 계층에서의 위임과 다단계 위임을 지원한다. 그리고 역할이 가진 속성과 사용자의 속성을 비교하고, 역할이 가진 제약을 적용함으로써, 위임역할에 지정될 수 있는 대상이 되는 사용자를 제한하여 비밀성(정보유출)과 무결성(의무분리)을 유지한다.

현재의 역할 기반 시스템에서는 역할에 사용자를 지정하는 일을 보안관리자가 주로 담당하고 있다. 이는 역할에의 사용자 지정에 보안관리자가 계속적으로 참여함으로써 특히 분산 환경에서 관리의 어려움을 증가시킬 수 있다. ERBAC 상에서 사용자 수준의 위임은 이러한 어려움을 극복할 수 있을 뿐 아니라, 분산 환경에 RBAC을 적용할 수 있는 수단을 제공한다.

참고문헌

1. 김학범, 홍기융, 김동규, 확장된 역할기반 접근통제 모델, 통신정보보호학회논문지 제9권 1호, pp.47-93, 1999. 3.
2. Longhua Zhang, Gail-Joon Ahn, Bei-Tseng Chu. A Rule-Based Framework for Role-Based Delegation. Proceedings of 6th ACM Symposium on Access Control Models and Technologies, Chantilly, Virginia, May 3-4, 2001.
3. Martin Abadi, Michael Burrows, Butler Lampson and Gordon Plotin. A calculus for Access Control in Distributed Systems. ACM Transaction on Programming Languages and Systems, Vol.15 No.4, Sept 1993, pp. 706-734
4. Ezedin Barka and Ravi Sandhu. A Role-based Delegation Model and Some Extensions. Proceedings of 16th Annual Computer Security Application Conference, Sheraton New Orleans, Dec. 11-15, 2000
5. Ezedin Barka and Ravi Sandhu. Framework for Role-Based Delegation Model. Proceedings of 23th National Information Systems Security Conference, pp. 101-114, Baltimore, Oct. 16-19, 2000
6. Henry M. Glad. Access Control for Large Collections. ACM Transactions on Information Systems, Vol.15, No.2, April 1997, pp. 154-194
7. J. Linn, M. Nystrom. Attribute Certification : An Enabling Technology for Delegation

and Role-Based Controls in Distributed Environments. ACM Workshop on RBAC 1999: 121-130

8. 심재훈, 박석, 역할기반 접근제어에 기초한 사용자 수준의 위임기법, 통신정보보호학회 논 문지 제10권 3호, pp.49-62, 2000. 9.

User-Level Delegation in Extended Role-Based Access Control Model

Chong-Hwa, Park*

Abstract

In current role-based systems, security officers handle assignments of users to roles. This may increase management efforts in a distributed environment because of the continuous involvement from security officers. The role-based delegation provides a means for implementing RBAC in a distributed environment. The basic idea of a role-based delegation is that users themselves may delegate role authorities to other users to carry out some functions on behalf of the former. This paper presents a user-level delegation model, which is based on Extended Role-Based Access Control(ERBAC). ERBAC provides finer grained access control on the base of subject and object level than RBAC model.

* Dept. of Software Semyung University