리눅스 서버 환경에서 RBAC 관계정보 관리를 위한 일관성 특성

(The consistency properties for management of RBAC relationship informations on the LINUX server environments)

오 석 균* 김 성 열**
(Sug-Kyun Oh) (Seong-Ryeol Kim)

요 약

역할기반 접근제어(RBAC)는 접근제어 정책의 관리단가를 낮출 수 있는 접근제어 메커니즘이다. RBAC 모델을 위해 개발된 관리도구(Admin Tool)는 사용자와 역할의 관계정보를 관리한다. 이들 관계에 대한 정보의 일관성을 유지하기 위해서 관계정보의 일관성을 정의하는 특성 집합이 필요하다. 본 논문은 Linux 서버 시스템 환경에서 RBAC 기술을 이용한 보안 시스템을 설계할 때 사용자와 역할의 관계에 관한 정보관리를 위해 관계정보 일관성 특성에 관하여 다루었다. 이는 더 효율적인 관리도구를 구현하기 위해 일관성 특성에 대해 동등한 결과를 얻을 수 있는 최소 집합의 개발을 유도하기 위함이다.

Abstract Role-Based Access Control(RBAC) is an access control mechanism that reduce the cost of administering access control policies. The Admin Tool developed for RBAC Model manages relationship informations of user and role. In order to maintain the consistency of the information for these relationships, a set of properties defining consistency of the relationship informations is required. When it will be designed security systems applying RBAC policy on the Linux server system environments, this paper described consistency properties of relationship informations for information management of user and role relationships. It leads us to the development of minimal set obtainable the equivalent results of consistency properties for a more efficient Admin Tool implementation.

1. 서 론

RBAC의 핵심은 허가(permission)가 역할(role)과 관련되고, 사용자는 적절한 역할에 할당된다는 것이다. 이 개념은 허가 관리를 매우 단순화 시켰으며, 역할이 조직 내에서 다양한 작업의 기능에 따라 생성되고 사용자의 책임과 자격을 근거로 사용자에 할당된다[1],[2]. 역할은 특정 임무에서 하는 능력을 나타낼 수도 있고, 권한과 책임을 구체적으로 명기할 수 있어 여러 사용자에게 특정 임무를 순환 시켜가며 할당할 수 있다. RBAC는 모델 설계와 구현에 역할의 모든 개념을 편리하게 수용할 수 있다[3]. 또한 RBAC는 접근제어 정책의 관리가 보다 쉽고 편리한 접근제어 메커니즘이다[4].

NIST에서 RBAC 모델을 위해서 개발한 관리도구(Admin Tool)는 사용자와 역할의 관계를 데이

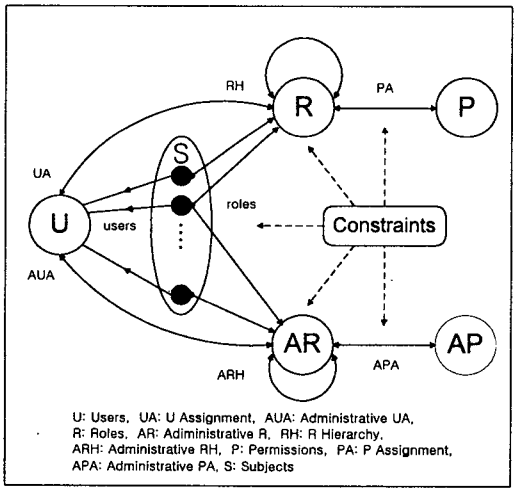
* 충청대학 컴퓨터학부 교수
** 청주대학교 컴퓨터정보공학과 교수

터베이스에 저장 및 관리한다. 이들 관계에 대한 데이터베이스 정보를 유지하는데 있어 데이터의 일관성 원칙이 이루어져야 한다. 따라서 이들 관계에 대한 데이터 일관성을 정의하는 특성 집합이 필요하다.

본 논문에서는 Linux 서버 시스템 환경에서 RBAC 기술을 이용한 보안 시스템을 설계할 때 사용자와 역할의 관계에 관한 정보 관리를 무결성 있게 유지하기 위한 데이터의 일관성 특성에 관하여 다루었으며, 관리도구는 NIST에서 개발한 것이고, 역할기반 접근제어 모델은 Sandhu 등이 제안한 RBAC96 모델을 기초로 하였다.

2. RBAC96 모델

Sandhu에 의해 제안된 RBAC 모델을 RBAC96이라 부르며, 이 모델군중에서 가장 일반적으로 많이 사용하는 모델이 $RBAC_3$로 <그림 1>과 같다[5].



<그림 1> RBAC96 모델

<그림 1>에서 상위 부분은 데이터와 자원에서 액세스를 통제하는 정규 역할과 정규 허가를 담당하고, 하위 부분은 관리 역할과 관리 허가를 담당한다.

역할은 순서쌍으로 표현하며, 반순서 관계를 갖는다. x>y이면 역할 x는 역할 y의 허가를 상속하고 x의 멤버는 y의 멤버를 내포한다. 이 경우 x는 y의 상급(senior)이라고 말한다. 이러한 RBAC96 모델의 구성요소들을 정의하면 <정의 1>과 같다.

<**정의** 1> RBAC96 모델의 구성요소

- 집합
  - U ; 사용자 집합
  - R ; 정규 역할이 서로 소인 집합
  - AR ; 관리 역할이 서로 소인 집합
  - P ; 정규 허가가 서로 소인 집합
  - AP ; 관리 허가가 서로 소인 집합
  - S ; 세션의 집합
- 사용자와 역할 관계
  - $UA \subseteq U \times R$; 역할에서 사용자로의 할당관계
  - $AUA \subseteq U \times AR$; 관리 역할과 사용자간의 다대다 할당관계
- 역할과 허가 관계
  - $PA \subseteq P \times R$; 역할에서 허가로의 할당관계
  - $APA \subseteq AP \times AR$; 관리 역할과 허가간의 다대다 할당관계
- 역할 계층
  - $RH \subseteq R \times R$; 순서쌍 역할 계층
  - $ARH \subseteq AR \times AR$; 순서쌍 관리 역할 계층
- 세션과 사용자 매핑
  - $user : S \rightarrow U$; 단일 사용자 $user(s_i)$에 각 세션 s_i를 매핑한다.
- 역할 또는 관리 역할과 세션 매핑
  - $roles : S \rightarrow 2^{R \cup AR}$; roles와 admin $roles(s_i) \subseteq \{r \mid (\exists r' \geq r)[(user(s_i), r') \in UA \cup AUA]\}$에 각 세션 s_i를 매핑한다.

세션 s_i는 $\bigcup_{r \in roles(s_i)} \{p \mid (\exists r'' \geq r)[(p,r'') \in PA \cup APA]\}$인 허가를 갖는다.

- 제약집합

여러 구성요소 값을 결정하여, 각 요소에 제약을 가하고, 받아들일 수 있는 값만 허용.

3. 직무 분리

시스템 관리자가 RBAC 메커니즘을 수행하기

위해서는 직무가 분리되어야 한다. 여러 작업들이 갖고 있는 관련된 능력들이 상호협력 처리하다보면 예기치 못했던 사건이 발생할 수 있는데, 이러한 사건을 막기 위해서는 직무를 분리해야만 한다. 직무분리 목적으로는 트랜잭션의 부분 집합에서 한 사람이 그 집합 내의 모든 트랜잭션을 수행하는 것을 허용하지 않아야 한다는 것이다[6]. 시스템 관리자는 관리 방법을 자연스럽게 추상적인 개념 수준에서 접근을 제어할 수 있어야 하고, 역할, 역할계층, 관계, 제약을 정의하고 수립함으로서 사용자의 활동을 통제함으로서 이룰 수 있다.

사용자들은 여러 가지 업무를 처리하면서 단독으로 처리되는 것과 여러 업무가 복합적으로 상호 보완하면서 처리되는 것이 있다. 따라서 사용자에게 역할의 할당과 사용하는 시기를 조절하여 직무를 분리하며, 정적 직무분리(ssd)와 동적 직무분리(dsd)가 있다.

ssd는 역할이 사용자에게 고정으로 할당되어 단독으로 처리하므로 다른 사용자에게 그 역할이 할당될 수 없는 경우이다. dsd는 사용자가 많은 역할을 할당받아 다양한 처리 과정에 많은 역할이 수시로 상호 협력하면서 동작하게 된다[7].

또한 dsd의 경우는 사용자에게 여러 역할이 할당되어 서로 교차 진행하면서 상호 협력하는 경우와 동시에 진행되면서 상호 협력하는 경우가 있다. 이러한 경우, 사용자가 여러 역할을 교차해 가면서 상호 협동하는 경우는 어느 한 역할을 사용할 때에 다른 역할을 같이 사용할 수 없기 때문에 상호 배타적(mutually-exclusive)이라고 하고 상호 배타적 직무분리(msd)라고 한다. 따라서 여러 역할을 동시에 수행하면서 상호 협동하는 경우는 서로 유기적인 관계를 갖고 동작하므로 보안상의 문제가 없는 한 제한을 가하지 않고 비교적 자유롭게 처리할 수 있게 해야하기 때문에 개방적(liberal)이라고 개방적 직무분리(lsd)라고 한다.

4. 관리도구(Admin Tool)

NIST에서 개발한 관리도구는 Sandhu 등이 제안한 $RBAC_3$을 웹 상에서 구현할 수 있게 개발된 관리도구로서 역할, 역할 계층, ssd, msd, 역할 빈도를 위해 사용자 권한을 관리한다[8]. 오류를 줄이기 위해서 관리도구는 역할 할당 개념을 사용하는데 이는 관리자가 조직을 나타내는 역할 계층에 대한 인식을 유지하는데 도움이 된다. 관리도구는 한 사용자에 이미 할당된 역할에서 상속받은 역할이 다른 사용자에게 할당되는 것을 허용하지 않는다. 또한, 두 역할이 msd 관계이고, 한 역할이 사용자에 권한이 부여되었으며, 한 역할이 다른 역할을 상속한다고 가정하면, 세션 내에 사용자의 ARS가 성립되었을 때, 명백한 모순이 발생한다. 이는 한 역할이 다른 역할을 상속하므로 다른 역할은 사용자의 ARS 내에 속하게 된다. 그러나 두 역할이 msd 관계이므로 다른 역할이 사용자의 ARS에 속할 수 없다[9].

이러한 문제를 해결하기 위해서 관리도구는 계층 관계와 msd 관계 양쪽을 동시에 가지는 역할쌍을 허용하지 않는다. 이러한 설계 목적은 관리 임무를 간략히 하기 위함이다.

5. 관계정보 일관성 특성

5.1 기본 집합과 기능

관계정보의 일관성 특성에 대한 기본적인 집합과 기능들은 다음과 같다.

- USERS : 사용자 집합
- ROLES : 역할 집합
- OPERATIONS : 사용자, 역할, 할당, 상속, ssd, msd, 역할 원리, 활동 중인 역할에 대한 설정, 추가, 삭제 등의 동작들이다.
- assigned_roles : $USERS \rightarrow 2^{ROLES}$.

assigned_roles(u)는 사용자 u에 할당된 역할 집합이다.

- active_roles : $USERS \rightarrow 2^{ROLES}$.

active_roles(u)는 사용자 u의 세션에서 ARS이다.

- inherits $\subseteq$ ROLES $\times$ ROLES.

역할간의 상속관계를 나타내는 것으로 $\rightarrow^a$는 asymmetric 상속, $\rightarrow^t$는 transitive 상속, $\rightarrow^r$은 reflexive 상속을 표시한다.

- ssd $\subseteq$ ROLES $\times$ ROLES.

역할간의 ssd(정적 직무분리) 관계를 나타낸다.

● msd ⊆ ROLES × ROLES.

역할간의 msd(상호 배타적 직무분리) 관계를 나타낸다.

● cardinality : ROLES→N∪{∞}.

cardinality(r)는 역할 r을 위해 권한이 부여된 사용자 최대 수를 의미한다.

5.2 유도과정

유도과정은 지정한 사용자를 위해 권한이 부여된 역할을 전달과정과 지정한 역할을 위해 권한이 부여된 사용자를 전달과정은 다음과 같이 유도된다.

먼저, 지정한 사용자를 위해 권한이 부여된 역할의 전달과정을 유도하면 다음과 같다.

● authorized_roles : USERS→2^{ROLES}

∀r∈ROLES, ∀u∈USERS,

u∈authorized_users(r) ⇔ ∃p∈ROLES

여기서 p∈assigned_roles(u)∧p→^{tr}r.

또한, 지정한 역할을 위해 권한이 부여된 사용자를 전달과정을 유도하면 다음과 같다.

● authorized_users : USERS→2^{ROLES}.

∀r∈ROLES, ∀u∈USERS,

u∈authorized_users(r) ⇔ r∈authorized_roles(u).

5.3 상태와 전이

● STATES는 상태들의 집합

● 상태는 USERS, ROLES, assigned_roles, active_roles, inherits, ssd, dsd, msd, 역할 원리들로 구성된 쌍이다.

● 상태 전이는 관리동작을 수행하는 관리자에 의해서 또는 사용자의 RBAC 세션동안에 역할을 취하거나 제거하는 사용자에 의해 시작된다.

● 전이 함수(δ)는 부분함수이다.

STATES×OPERATIONS×2^{ARGS} → STATES

여기서 ARGS는 인수의 개수이다.

5.4 일관성 특성

사용자와 역할의 관계정보는 다음과 같이 정의한 특성들을 만족하여야 일관성 있게 관계정보를 유지할 수 있다.

<특성1>

역할 r에 권한이 부여된 사용자 수는 그 역할의 빈도를 초과하지 않아야 한다.

∀r∈ROLES,

|authorized_users(r)|≤cardinality(r).

<특성2>

한 역할이 자신의 역할로 상속하지 않아야 한다. 즉, 비반사적(irreflexive)이어야 한다.

∀r∈ROLES, ¬(r→^{r}r).

<특성3>

동일한 사용자에 할당된 두 역할이 lsd 관계가 아닐 때는 서로 상속관계가 성립하지 않는다. 즉, 비대칭적(asymmetric)이어야 한다.

∀r_1, r_2∈ROLES, ∀u∈USERS, r_1,r_2 ∉lsd,

r_1, r_2∈assigned_roles(u) ⇒ (r_1→^{a}r_2).

<특성4>

ssd 관계에 있는 두 역할은 동일한 사용자에 권한이 부여되지 않아야 한다.

∀r_1, r_2∈ROLES, ∀u_1, u_2∈USERS,

r_1∈authorized_roles(u_1), r_2∈authorized_roles

(u_2), (r_1, r_2)∈ssd ⇒ u_1≠u_2.

<특성5>

역할은 자기자신의 역할과 ssd 및 msd 관계를 성립하지 않는다. 즉, 비반사적이어야 한다.

∀r∈ROLES ⇒ (r, r) ∉ssd and msd.

<특성6>

두 역할의 순서쌍이 ssd(msd) 관계이면 반대 순서쌍도 ssd(msd) 관계로 대칭적(symmetric)이다.

6-1) $\forall r_1, r_2 \in ROLES, (r_1, r_2) \in ssd \Rightarrow (r_2, r_1) \in ssd.$

6-2) $\forall r_1, r_2 \in ROLES, (r_1, r_2) \in msd \Rightarrow (r_2, r_1) \in msd.$

<특성7>
한 역할이 다른 역할을 상속하면 이 두 역할은 ssd 및 msd 관계가 성립하지 않는다.

$\forall r_1, r_2 \in ROLES, r_1 \rightarrow^a r_2 \Rightarrow (r_1, r_2) \notin ssd \text{ and } msd.$

<특성8>
한 역할로부터 두 역할이 상속되었다면 상속된 두 역할은 ssd 및 msd 관계를 이루지 않는다.

$\forall r, r_1, r_2 \in ROLES, r \rightarrow^a r_1, r \rightarrow^a r_2 \Rightarrow (r_1, r_2) \notin ssd \text{ and } msd.$

<특성9>
한 역할이 다른 역할을 상속하고 상속된 역할이 또다른 역할과 ssd(msd) 관계를 이루면 상속한 역할도 또다른 역할과 ssd(msd) 관계를 이룬다.

9-1) $\forall r, r_1, r_2 \in ROLES, r \rightarrow^a r_1, (r_1, r_2) \in ssd \Rightarrow (r, r_2) \in ssd.$

9-2) $\forall r, r_1, r_2 \in ROLES, r \rightarrow^a r_1, (r_1, r_2) \in msd \Rightarrow (r, r_2) \in msd.$

<특성10>
사용자 u의 ARS는 그 사용자에 권한이 부여된 역할의 부분집합이다.

$\forall u \in USERS, active\_roles(u) \subseteq authorized\_roles(u).$

<특성11>
두 역할간에 msd 관계가 성립하면 두 역할은 동일한 사용자에 있는 ARS가 아니다.

$\forall r_1, r_2 \in ROLES, \forall u_1, u_2 \in USERS, r_1 \in active\_roles(u_1), r_2 \in active\_roles(u_2), (r_1, r_2) \in msd \Rightarrow u_1 \neq u_2$

<특성12>
두 역할이 msd 관계이면 이들 역할은 ssd 관계를 이루지 않으며, 그 반대도 성립하지 않는다. 즉 msd와 ssd 관계는 서로 교차하지 않는다.

12-1) $\forall r_1, r_2 \in ROLES, (r_1, r_2) \in msd \Rightarrow (r_1, r_2) \notin ssd$

12-2) $\forall r_1, r_2 \in ROLES, (r_1, r_2) \in ssd \Rightarrow (r_1, r_2) \notin msd.$

<정의 2>
RBAC에서 사용자와 역할에 관한 관계정보가 한 상태에서 일관성을 갖기 위한 필요충분 조건은 그 상태가 <특성1>~<특성12>까지를 만족하여야 한다.

6. 결 론

관리도구는 사용자와 역할의 관계정보를 저장 및 관리한다. 이들 관계에 대한 정보를 유지하는데 있어 데이터의 무결성이 보장되어야 한다. 데이터 무결성 보장을 위해서는 관계정보에 대한 일관성이 있어야 한다. 따라서 이들 관계에 대한 관계정보 일관성을 정의하는 여러 특성 집합이 필요하다.

본 논문에서는 Linux 서버 시스템 환경에서 RBAC 기술을 이용한 보안 시스템을 설계할 때에 사용자와 역할의 관계에 관한 정보관리를 위해 관계정보 일관성 특성을 정의하였다. 이러한 특성 정의에 관한 연구로 효과적인 관리도구를 구현할 때에 일관성 특성과 결과에 대해 동등한 효과를 얻을 수 있도록 축소된 집합의 개발을 유도할 수 있게 되었다.

연구결과, RBAC는 사용자와 역할의 관계정보에 대한 일관성 특성의 최소 집합을 찾아냄으로써 더욱 뛰어난 성능을 갖게 될 것이다.

향후 연구로는, 관리도구의 일관성 검사를 위한 알고리즘의 개발과 이에 따른 알고리즘 성능분석에 대한 연구가 진행되어야 한다.

참고문헌

[1] David Ferrailo and Richard Kuhn : Role-Based Access Control, Proceedings of 15th National Computer Security Conference, Oct. 1992, pp.554-563.

[2] Ravi S. Sandhu, Edward J. Coyne, Hal L. Feinstein and Charles E. Youman : Role-Based Access Control: A Multi-Dimensional View, Proc. of 10th Annual Computer Security Applications Conf., Dec. 1994, pp.54-62.

[3] Ravi S. Sandhu, and Venkata Bhamidipati : The URA97 Model for Role-Based User-Role Assignment, Proc. of IFIP WG 11.3 Workshop on Database Security, Aug. 1997.

[4] D. Ferraiolo, J. Cugini, and D.R. Kuhn : Role Based Access Control: Features and Motivations, In Annual Computer Security Applications Conference, 1995.

[5] Ravi S. Sandhu, Edward J. Coyne, Hal L. Feinstein and Charles E. Youman : Role-Based Access Control Models, IEEE Computer, Vol. 29, No. 2, Feb. 1996, pp.38-47.

[6] Ravi. S. Sandhu, Seperation of duties in computerized information systems, In S. Jajodia and C. E. Landwehr, editors, Database Security IV : status and Prospects, pp.179-189, North-Holland, 1991.

[7] John. F. Barkley, D. Richard Kuhn, Lynne S. Rosenthal, Mark W. Skall and Anthony V. Cincotta : Role Based Access Control for the Web, CALS Expo International & 21st century commerce 1998 : Global business solutions for New Millennium, 1998.

[8] John. F. Barkley, Anthony V. Cincotta, David F. Ferraiolo, Serban Gavrilla and D. Richard Kuhn : Role Based Access Control for the World Wide Web, 20th national computer security conference, April. 1997.

[9] Serban I. Gavrila, John. F. Barkley, : Formal Specification for Role Based Access Control User/Role and Role/Role Relationship Management, Proceedings of the Third ACM Workshop on Role-Based Access Control, October, 1998.