역할기반 접근제어 시스템 환경에서 무결성 보장을 위한 세션기반 응용 프로그램의 설계 및 운영 구조

이 형효, 최 은복, 노 봉남
전남대학교 전산학과

A Framework for Session-based Integrity Enforcement Application Design and Operation in Role-Based Access Control Systems

HyungHyo Lee, EunBok Choi, BongNam Noh
Dept. of Computer Science, Chonnam National University

요 약

역할기반 접근제어 정책은 정보보호를 위한 단순한 접근제어 수단으로서 뿐만 아니라, 사용자와 다양한 응용 소프트웨어로 구성된 환경에서 편리한 권한 관리 기능을 수행할 수 있어 일반적인 기업환경과 컴퓨터 운영체제, 데이터베이스와 같은 시스템에서 채택, 활용되고 있다. 본 논문에서는 역할기반 접근제어 정책을 채택한 정보 시스템 환경에서 시스템이 관리하는 정보의 무결성 보장을 위한 세션기반 임무분리 특성을 제안하고, 정형적 기술방식을 Z 언어를 이용하여 표현하였다. 또한 제안된 특성을 기반으로 한 응용 프로그램 명세 요소를 이용한 응용 프로그램 설계 및 실행가능성 확인기능, 전체 시스템의 운영 구조에 대하여 기술하였다.

1. 서 론

컴퓨터 시스템과 정보시스템들이 네트워크를 통해 서로 연결되고 각 기관간의 정보 공유와 신속한 유통이 촉진됨에 따라 정보 보안에 대한 중요성도 한층 증가되었다. 이에 미국 국방성은 1985년 중요 정보의 보안성질과 정책 등을 규정한 TCSEC(Trusted Computer System Evaluation Criteria)을 제정하였고[5], 유럽에서는 정보보호시스템의 평가를 위한 ITSEC(Information Technology Security Evalutation Criteria)을 제정하였다.

TCSEC은 자율적 접근제어(DAC: Discretionary Access Control)와 강제적 접근제어(MAC: Mandatory Access Control) 등의 2가지 접근제어 정책에 대하여 규정하고 있다. 강제적 접근제어 정책은 각 정보에 결합된 비밀 등급(classification level)과 사용자에 부여된 인가 등급(clearance level)을 미리 규정된 규칙에 의해 비교하여 그 규칙을 만족하는 사용자만에게 접근권한을 부여하는 보안정책으로서, 군사 환경과 같은 정보의 기밀성(confidentiality)이 매우 통제된 환경에서만 주로 사용되고 있다[4, 17]. 이에 반해 자율적 접근제어 정책은 정보 소유자의 자율적 판단에 의해 정보에 대한 접근권한이 결정되는

보안 정책으로 강제적 접근제어 정책에 비해 매우 유연한 권한부여 기능을 제공한다. 그러나, 각 기업 환경은 다른 기업들과는 다른 보안 정책과 요구사항들을 가지고 있어서, TCSEC에서 규정된 자율적 접근 제어나 강제적 접근 제어 정책만으로 기업의 보안 요구를 만족시킬 수 없는 문제점을 가지고 있다. 또한 자율적 접근 제어 정책 역시 기업 환경에 적용하기에 적합하지 않다고 평가되고 있다[3, 15].

역할기반 접근제어(RBAC: Role-Based Access Control) 정책은 정보에 대한 사용자의 권한부여 여부를 각 사용자의 식별자나 이미 정의된 규칙에 의해 판단하지 않고, 각 사용자가 수행중인 해당 조직내에서의 역할(role)에 의해 결정하는 특징을 가지고 있다. 역할기반 접근제어 시스템에서는 정보에 대한 접근 권한이 사용자에게 직접 부여되지 않고, 조직에서 규정된 역할들에게 배정된다. 또한 사용자는 임의적으로 접근 권한을 다른 사용자에게 위임하거나 다른 사용자에게 할당된 접근 권한을 철회할 수 없으며, 사용자가 정보에 대한 연산(operation)을 수행하기 위해서는 주어진 정보에 대한 연산을 수행할 수 있는 역할에 구성원(member)이 되는 방법밖에 없다.

이러한 역할기반 접근제어 정책은 정보에 대한 통제가 제한된 수의 관리자에 의해 수행하고, 각 사용자마다 접근 권한을 배정하여 관리하는 대신 조직내에서의 역할에 따라 접근 권한을 부여하는 방법이 사용자가 대단히 많은 실제 기업 환경에 효과적으로 적용할 수 있어 이에 대한 활발한 연구가 현재 진행중이다[6, 9, 11].

주어진 기업 환경에서 수행되는 업무는 자료의 무결성 보장을 위한 해당 조직의 규정에 의해 한 명 이상의 사용자에 의해 수행된다. 따라서 조직에서 관리하는 정보의 무결성에 영향을 미치는 응용 프로그램을 수행하는 사용자들의 접근제어가 필수적이다. 이를 위하여 몇 가지의 임무분리(SOD: Separation of Duty) 정책이 정의되었고, 임무분리 정책들은 그 정책이 집행되는 시점(정적, 동적), 유연성, 임무분리 대상(역할, 객체) 등에 따라 구분될 수 있다[2]. 임무분리 정책은 정보의 무결성과 관련된 연산들을 여러 역할이나 사용자에게 분산시킴으로써 조직에서 관리하는 정보의 무결성 침해 가능성을 최소화하는데 있다. 현재까지 진행된 임무분리 정책과 그와 관련된 연구는 주로 임무분리의 종류와 관계, 그리고 정형적인 기술(formal description)을 중심으로 진행되었다[1, 3, 12].

그러나, 실제 기업 환경에서 역할기반 접근제어 정책과 임무분리 정책을 함께 적용할 때, 이를 기반으로 한 응용 프로그램의 설계나 운영방법 등에 대한 연구는 미미한 실정이다. 또한 역할기반 접근제어 모델의 구성요소로서 응용 프로그램의 실행 단위인 세션(session)에 대한 정의와 특성에 대한 연구가 부족하여 실제 시스템의 설계와 운영에 많은 문제점을 가지고 있다.

따라서 본 논문에서는 응용 프로그램의 실행 단위인 세션을 특성과 세션을 기반으로 한 임무분리 특성, 세션들의 관련성에 기반을 둔 응용 프로그램의 설계 및 운영 구조에 대해 기술한다.

2. 역할기반 접근제어 정책

TCSEC에 규정된 강제적 접근제어 정책은 군사 환경이나 매우 제한적인 도메인에서 매우 제한된 수의 보안 관리자(security administrator)들에 의해 정의된 일정한 규칙에 따라 정보에 대한 사용자의 접근 권한을 제어한다. 이에 반하여, 자율적 접근제어는 각 정보의 소유자들이 소유자들의 자율적인 판단에 따라 그 정보에 대한 접근 권한을 다른 사

용자에게 위임하거나 취소할 수 있는 권한을 가지고 있어, 강제적 접근제어 정책보다 정보에 대한 훨씬 유연하고 분산된 접근제어 기능을 수행할 수 있는 특징을 가지고 있다. 그러나 이 두 정책 모두 실제 기업환경에 적용되기에는 부적합한 특성들이 있다. 예를 들면 기업환경에서는 강제적 접근제어 정책에서 중요시 하고 있는 정보의 기밀성 못지 않게 무결성을 중요시 하고 있으며, 기업내 대부분 정보의 소유자가 기업 소속원이 아닌 몇 명의 관리자들에게 소속되는 점 등이 있다[3].

역할기반 접근제어 정책은 기업 환경 뿐만 아니라 데이터베이스, 운영체제 등에 적용될 수 있는 매우 유연한 접근제어 정책으로[7], 자율적 또는 강제적 접근제어 정책보다 정보에 대한 추상적인 접근통제와 효율적인 접근권한 관리를 수행할 수 있는 장점을 가지고 있다[12, 13].

Sandhu 등은 역할기반 접근제어 모델을 특성별로 정의하였고, Ferraiolo 등은 역할기반 접근제어 모델에 대한 정형적인 기술과 함께 역할기반 접근제어 정책을 기반으로 한 시스템의 구조와 프로토타입에 대한 연구를 수행하였다[1, 3].

역할기반 접근제어 정책은 정보의 기밀성 보장이나 정보의 소유자에 의해 접근권한의 부여되는 등의 이미 정의된 정책만을 고정적으로 지원하는 대신, 주어진 기업 환경의 특성과 필요에 따라 유연한 접근권한 정책을 구현할 수 있어 정책에 독립적(policy-neutral)인 특성을 제공하고 있다[7, 8, 10, 16]. 이 절에서는 역할기반 접근제어 모델의 주요 특성에 대해 살펴본다.

2.1 특성 및 구성요소

역할기반 접근제어 모델에서는 정보에 대한 연산을 수행할 수 있는 권한(permission, privilege)들은 사용자에게 직접 할당되지 않고, 주어진 기업 환경에서 정의된 역할에 대해서만 배정되는 특징을 가지고 있다. 따라서 사용자가 원하는 정보에 대한 연산을 수행하기 위해서는 먼저 해당 정보에 대한 연산을 실행할 수 있는 권한을 가진 역할의 소속원이 되어야 한다. 사용자를 특정 역할의 소속원으로 배정하는 권한은 미리 정해진 보안 관리자들에 의해 수행될 수 있다. 따라서 전체 시스템의 보안 관리를 위한 제어가 몇 명의 보안관리자에 의해 이루어지게 되므로, 자율적 접근제어 정책에서 발생할 수 있는 접근권한 통제의 어려움을 해결할 수 있다.

이처럼 역할기반 접근제어 모델에서는 권한의 관리(permission management)를 사용자와 정보 객체간의 관계로 인식하는 대신 기업 환경내에서의 역할과 정보 객체간의 관계로 설정, 관리함으로써 사용자와 정보 객체의 수가 대단히 많은 실제의 기업 환경에 매우 적합한 특성을 제공한다[3, 12]. 그리고 역할들간의 관계는 현재 기업 환경의 의사결정 체계와 관리체계에 잘 부합되어 일반적인 기업 환경의 권한체계들이 역할기반 접근제어 모델로 쉽게 모델링되는 장점이 있다.

또한 최소권한 원칙(least privilege principle), 임무분리, 자료 추상화(data abstraction)와 같은 주요 보안 원칙들 역시 지원하고 있다[3]. 그림 1은 역할기반 접근제어 모델의 구성요소를 보여주고 있다.

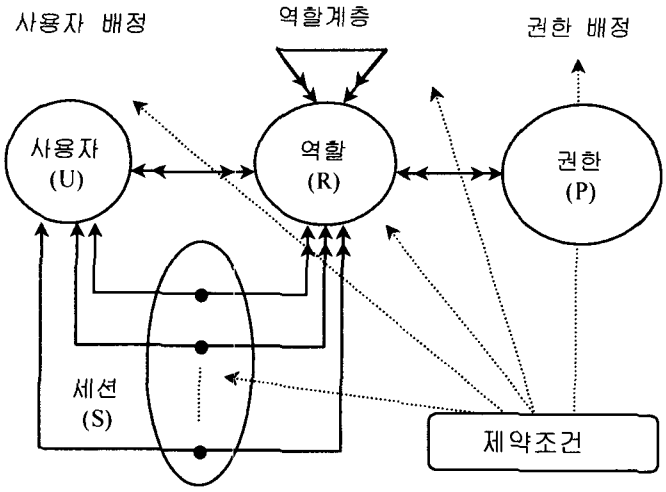

그림 1 역할기반 접근제어 모델의 구성요소

■ 역할, 역할계층, 사용자, 권한, 세션

역할(role)은 주어진 기업 환경내에서 정의된 업무의 기능으로서, 각 역할이 수행가능한 권한과 함께 그에 따른 책임으로 구성된다. 역할에 배정된 권한은 조직의 규정이나 규칙에 의해 정의되며, 역할에 소속된 사용자들에게 동일하게 제공된다. 역할 계층(role hierarchy)은 관련성 있는 역할들간의 부분순서(partial order) 관계로서 기업의 권한과 책임 체계와 매우 유사하여 기업의 권한체계를 모델링하는데 매우 유용하게 사용된다. 사용자(user)는 컴퓨터 시스템을 통하여 시스템내의 정보를 사용하는 객체로서 한 사용자는 한 명의 사람에 대응된다.

역할기반 접근제어 모델에서의 권한(permission)은 인가(authorization), 접근권한(access right) 그리고 특권(privilege)과 같은 의미를 가지고 있으며, 특정 정보 객체와 해당 객체에 대해 수행 가능한 연산의 집합으로 구성된다. 여기서 정보 객체는 기업내의 정보 시스템을 구성하고 있는 자료나 시스템 자원으로 구성된다.

세션은 한 사용자와 여러 개의 역할들의 집합으로 표현될 수 있으며, 사용자는 세션을 통하여 자신에게 배정된 역할들중의 일부 또는 전체를 수행할 수 있다. 이 때 세션에 의해 수행되는 역할들을 활성화된 역할(active role)이라 하며, 한 사용자는 여러 개의 세션을 실행할 수 있다.

■ 역할 배정, 권한 배정

역할 배정(role assignment)과 권한 배정(permission assignment)은 다대다 관계이며 역할기반 접근제어 모델에서 매우 중요한 구성요소이다. 역할기반 접근제어 모델의 가장 특징중의 하나는 사용자가 정보 객체들에 대해서 실행할 수 있는 연산들을 직접 사용자에게 부여하는 대신 조직의 업무 수행에 필요한 역할과 각 역할에 대한 권한을 배정하고(권한배정), 사용자는 해당 역할에 구성원이 됨으로써(역할배정) 정보객체에 대한 원하는 연산을 수행하도록 하는 것이다. 이러한 방법은 사용자와 정보 객체의 수가 많은 일반 기업 환경에서 권한의 관리를 매우 용이하게 수행할 수 있는 장점을 제공한다.

■ **제약조건**

제약조건(constraints)은 위에서 정의된 모든 구성요소에 대하여 적용될 수 있으며, 제약조건의 예로서는 임무분리, 한 역할에 할당될 수 있는 최대 사용자 수(cardinality), 사용자가 특정 역할의 소속원이 되기 위한 선수(prerequisite) 역할 등이 있다.

2.2 정형적 기술

정보 시스템에 대한 정형적인 기술은 소프트웨어 설계와 구현, 시험 등의 개발 전과정에서 보다 완벽한 시스템의 구축에 두움을 주며, 여러 개발자 사이에서 발생할 수 있는 시스템 사양이나 기능에 대한 잘못된 이해를 방지하는데 활용될 수 있다. 또한 정형적 기술방법은 시스템의 동작 특성을 명확히 기술하고 분석하는데 사용된다. 본 논문에서는 역할기반 접근제어 모델과 구성요소, 그리고 다양한 임무분리 특성을 기술하는데 Z 언어를 이용한다. Z 언어는 집합론과 predicate calculus를 기반으로 한 정형적 기술방법으로서, 시스템의 정적, 동적 특성을 표현하는 스키마를 이용하여 시스템의 전반적인 속성을 정형적으로 기술한다[18].

다음은 역할기반 접근제어 모델을 Z 언어를 이용하여 기술하는 데 필요한 Z 언어 타입들이다.

[*UESRS, OPERATIONS, OBJECTS, CONSTRAINTS*]
USERS: 사용자들의 집합
OPERATIONS: 연산들의 집합
OBJECTS: 정보를 포함하고 있는 객체 또는 시스템 자원
CONSTRAINTS: 제약조건들의 집합

위에서 정의된 기본 타입들을 이용하여 정의된 역할기반 접근제어 모델의 구성요소들의 상태 스키마는 다음과 같다.

Permission
objects: *OBJECTS*
operations: *OPERATIONS*

Role
name: *STRING*
users: *USER*
permissions: Permission

Session
name: *STRING*
user: *USERS*
roles: Role
operations: *OPERATIONS*
objects: *OBJECTS*

그리고, 다양한 임무분리 특성과 역할기반 접근제어 모델의 특성 기술을 위해 사용되는 함수들은 다음과 같다.

users_of_role: Role → ***USERS***
user_of_session: Session → ***USERS***
roles_of_session: Session → Role
active_roles: session → Role
mutex_roles: Role ∪ Role → Role

3. 임무분리 정책

군사 환경이나 일부 기관에서 정보의 기밀성을 중요시하는 반면, 일반 기업환경에서는 정보의 무결성이 매우 중요한 보안 속성들중의 하나이다. 이처럼 정보 시스템의 무결성 보장을 위하여 임무분리 특성이 정의되었고, 이를 통하여 몇몇 사용자들의 부정으로 인한 시스템의 무결성 손상 가능성을 최소화하는데 그 목적이 있다[2]. 따라서 정보 시스템의 무결성에 손상을 입힐 수 있는 권한을 가진 역할들, 즉 상호배타적인 역할(mutually exclusive roles)들을 그 역할들이 수행하는 권한들간의 연관성에 따라 동일 사용자에게 할당하지 않음으로써 무결성 침해의 가능성을 줄이게 된다.

정적 임무분리(static separation of duty)는 동일한 사용자를 상호배타적인 역할에 배정하지 않음으로써 정보 시스템의 무결성을 유지한다. 그러나 이 방법은 결과적으로 많은 사용자가 필요하여 시스템 운영에 따른 부담이 증가하고, 시스템 운영이 유연하지 못한 단점을 가진다.

동적 임무분리(dynamic separation of duty)는 사용자가 비록 상호배타적인 역할에 배정되더라도, 실제 세션에 의해 활성화되는 역할들이 상호배타적인 관계에 있지 않도록 유지하는 방법이다. 이 방식은 정적 임무분리 방법에 비해 시스템의 구성이 쉬어지고, 운영의 유연성이 증대되는 장점을 가지고 있다. 대신 실제 응용 프로그램이 동작할 때, 각 사용자가 세션을 통해 활성화하는 역할들간의 관계를 확인하여야 하는 과정이 추가로 요구된다.

정적, 동적 임무분리 외에 또 하나의 다른 임무분리 속성이 연산대상(operational) 임무분리이다. 이 속성은 정보 시스템 환경에서 수행되는 임의의 응용 프로그램에 대하여, 그 응용 프로그램에서 사용되는 모든 연산들이 한 사용자에 의해 수행되는 것을 방지되도록 기술하고 있다.

한편 다양한 임무분리 속성에 대한 정의와 특징, 정형적 기술에 대한 연구가 진행되었으며[13, 14], Gilgor 등은 지금까지 연구된 임무분리 속성들에 대한 관계에 대한 연구를 수행하였다[2].

정적 임무분리와 동적 임무분리 속성에 대한 스키마를 정의하면 다음과 같다.

StaticSOD

r1, r2: Role

$\forall$ r1, r2 $\in$ Role, r1 $\neq$ r2 $\cdot$
users_of_role(r1) $\cap$ users_of_role(r2) $= \varnothing$

dynamicSOD

r: Role
s?: Session

$\forall$ r $\subseteq$ active_roles(s?) $\cdot$
r $\cap$ mutex_roles(r) $= \varnothing$

정적, 동적 임무분리 특성을 정보 시스템 운영에 잘 적용하면 시스템의 무결성 유지에 효과적으로 활용될 수 있다. 운영 임무분리를 정형적으로 기술하기 위하여 응용 프로그램에 대한 정적 스키마를 먼저 정의할 필요가 있다.

응용 프로그램은 기업 환경에서 정보 시스템의 무결성을 유지시키면서 주어진 업무를 수행하는 단위로서 여러 개의 연관된 세션들과 세션들간의 제약조건들의 집합으로 기술될 수 있다. 응용 프로그램의 구성요소인 제약조건으로는 구성 세션간의 무결성 침해 가능성, 수행순서, 시간적인 제약 등이 포함될 수 있다. 또한 응용 프로그램은 미리 정의된 역할들에 의해서만 기동될 수 있으며, 고유한 식별자를 가지고 있다.

Application

name: ***STRING***
initiators: Role
sessions: Session
constraint: ***CONSTRAINTS***

initiators $\subseteq$ session.roles

OperationalSOD

application: Application
s: Session

$\forall$ s $\in$ application.sessions
#(**ran**(user_of_session(s))) $\neq$ 1

그러나 위에서 정의된 연산대상 임무분리 특성은 응용 프로그램을 구성하는 세션간의 특성과 연관성에 대한 고려가 포함되지 않아 비록 연산대상 임무분리 속성을 만족하더라도 시스템의 무결성이 손상되는 경우가 발생하게 되는 문제점을 가지고 있다. 본 논문에서는 이러한 문제점을 해결하기 위한 보다 엄격한 임무분리 특성을 4절에서 정의하고, 이러한 임무분리 특성을 이용한 응용 프로그램의 설계과 운영 구조에 대한 구조에 대해 기술한다.

4. 세션기반 응용 프로그램 설계 및 운용 구조

앞 절에서 정의한 바와 같이 응용 프로그램은 정보 시스템의 무결성을 유지시키면서 주어진 업무를 수행하는 단위로서 여러 개의 연관된 세션들과 세션들간의 제약조건들의 집합으로, 정보 시스템의 무결성 유지를 위하여 응용 프로그램을 대상으로 한 무결성 유지여부 확인 과정이 필수적이다. 다시 말하면, 응용 프로그램의 실질적 수행 기본단위인 세션에 대한 무결성 유지확인에 대한 점검 과정이 필요하다는 것이다.

그러나 현재까지는 세션에 대한 명확한 특성, 특히 세션을 대상으로 한 임무분리 속성에 대한 연구가 거의 이루어지지 않았다. 본 절에서는 역할기반 접근제어 정책을 기반으로 한 시스템의 구성과 운영에 필수적인 세션의 특성과 세션대상의 임무분리 속성, 세션의 연관성 기술의 분석을 통한 응용 프로그램의 실행가능성 확인과정 등에 대하여 기술한다.

4.1 응용 프로그램 설계 및 운영 구조

역할기반 접근제어 모델을 기반으로 구성된 정보시스템의 운영 구조는 보안 관리자에 의해 수행되는 역할기반 접근제어 모델의 구성요소 관리부문, 응용 프로그램 설계자에 의한 응용 프로그램 명세 및 실행가능성 확인부문, 그리고 일반 사용자들에 의해 설계된 응용 프로그램이 운영되는 부문으로 구성된다(그림 2).

정보시스템의 보안 관리자는 이미 정의된 임무분리 특성들(정적, 동적, 연산대상)을 기반으로 하여 역할기반 접근제어 모델의 구성요소의 변경기능을 수행한다. 응용 프로그램 설계자는 정보시스템의 무결성 보장을 위해 임무분리 특성들을 만족시키면서 보안 관리자에 의해 관리되는 사용자, 역할, 권한, 역할배정, 권한배정 등의 정보를 활용하여 새로운 응용 프로그램을 설계한다. 이때 설계된 응용 프로그램은 사용자에 의해 실행되기 전 현재의 역할기반 접근제어 모델의 구성요소를 이용하여 실행될 수 있는지 확인하는 절차를 거치게 된다. 정보 시스템 운영 단계에서 각 응용 프로그램은 시스템에 정의된 임무분리 특성, 역할기반 접근제어 모델의 구성요소 정보들을 참조하여 정보시스템의 무결성을 침해하지 않는 범위내에서 실행된다.

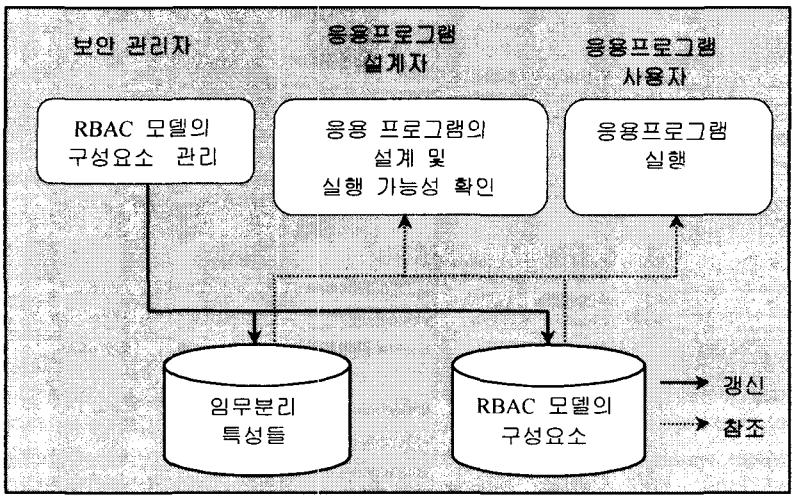

그림 2 역할기반 접근제어 모델 기반의 시스템 운영 구조

4.2 세션-사용자 배정, 세션-역할 배정

응용 프로그램 수행의 기본 단위인 세션은 한 명의 사용자와 그 사용자에 배정된 역할중의 일부 또는 전체로 구성된다[3]. 주어진 세션에 대하여 사용자를 배정하는 방법(SUA: Session-User Assignment)과 역할들을 배정하는 방법(SRA: Session-Role Assignment)은 그 배정 시기에 따라 시스템의 유연성과 동작성능에 많은 영향을 주는 요소들이다. 만일 응용 프로그램의 설계자가 SUA, SRA 과정을 응응 프로그램 설계 과정때 결정한다면, 시스템의 동작 성능의 개선될 수 있으나 시스템의 유연성을 감소하게 되며, SUA, SRA 과정을 실행시간에 결정한다면 시스템의 동작성능은 저하될 수 있으나, 시스템 운영의 유연성은 증가하게 된다.

본 논문에서 제안하는 SUA, SRA 과정의 실행시점은 SRA의 경우 응용 프로그램 설계과정때, SUA의 경우 응용 프로그램 실행때 결정하는 방식이다. 그 이유는 응용 프로그램을 구성하는 각 세션이 실행할 수 있는 권한들이 응용 프로그램의 기능에 의해 대부분 결정되어 있는 반면, 세션을 수행할 수 있는 사용자는 여러 명이고 응용 프로그램 실행때의 조건에 따라 그들중 한 명이 결정되는 방식으로 비교적 유동적이기 때문이다.

이와 같은 방식으로 SUA, SRA 과정을 실행하면 두 과정 모두를 응용 프로그램의 설계과정이나 실행과정때 실행하는 것보다 보다 개선된 시스템 운영의 성능 향상과 유연성을 제공할 수 있다.

4.3 세션기반 임무분리 특성

기존의 연산대상 임무분리 특성은 주어진 응용 프로그램에서 사용되는 모든 연산을 수행할 수 있는 사용자에게 응용 프로그램의 수행권한을 부여하지 않음으로써, 시스템의 무결성을 유지하는 데 그 목적이 있다. 그러나 어떤 경우에는 비록 연산대상 임무분리 특

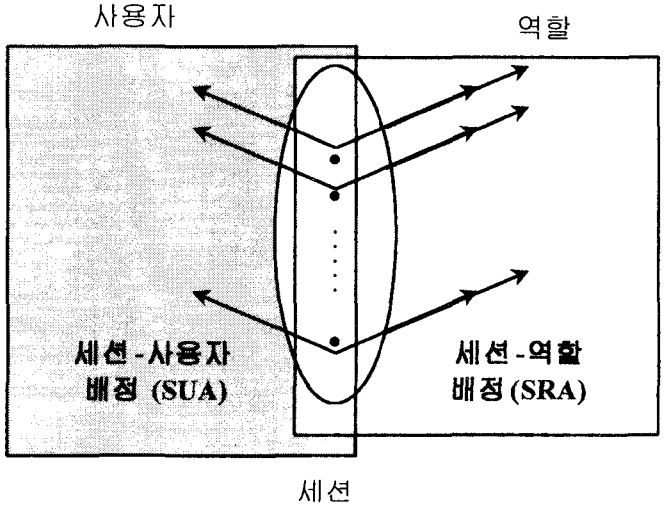

그림 3 세션-사용자 배정, 세션-역할 배정

성을 만족하더라도 시스템의 무결성이 침해받기 경우가 있어, 보다 엄격한 임무분리 특성의 정의가 필요하게 된다. 즉, 연산대상 임무분리 특성을 만족시키는 응용 프로그램에 대하여 그 응용 프로그램을 구성하는 세션들중의 일부가 한 사용자에게 할당되었을 때 무결성 침해가 발생될 수 있다. 예를 들어 다음과 같은 3개의 세션으로 구성된 응용 프로그램을 가정한다.

s_1: 물품구매 요청
s_2: 구매요청한 물품 종류와 수량 확인
s_3: 구매된 물품에 대한 대금지불 허가

만일 사용자 u_1이 세션 s_1, s_2를 수행하는 사용자로, 사용자 u_2가 세션 s_3를 수행하는 사용자로 결정되었다고 가정할 때, 비록 사용자 u_1이 응용 프로그램에서 사용되는 모든 권한을 수행할 권한이 없어도 물품구매 과정의 무결성을 침해할 가능성이 발생됨을 알 수 있다.

따라서, 본 논문에서는 응용 프로그램을 구성하는 세션을 기반으로 한 임무분리 특성을 제안한다. 일반적으로, 모든 응용 프로그램에 적용되는 세션기반의 임무분리 특성에 대한 정의는 불가능하기 때문에, 시스템의 특성에 따라 보안 관리자와 응용 프로그램 설계자에 의해 한 사용자에게 동시에 배정되어서는 않되는 상호배타적인 세션이 결정된다. 상호배타적인 세션은 응용 프로그램을 구성요소중 제약조건에 포함될 수 있으며, 그 정의에 대한 정형적 기술은 다음과 같다.

$$MUTEX\_SESSIONS\_SET == \{\ \text{Session}\ \}$$

MutexSessions

$$\begin{aligned}
&application: \text{Application}\\
&mutex_sessions: MUTEX_SESSIONS_SET\\
&\hline
&\forall\ session_set: \text{Session} \cdot (si, sj \in \text{Session} \cdot\\
&\quad session_set \in mutex_sessions \wedge\\
&\quad \#(session_set) > 2 \wedge si \neq sj \wedge\\
&\quad user_of_session(si) \neq user_of_session(sj))
\end{aligned}$$

4.4 실행가능성 확인 스키마

응용 프로그램 설계자에 의하여 설계된 응용 프로그램이 세션기반 임무분리 특성을 만족시키면서 실제 운영환경에서 실행될 수 있는지 확인하는 과정은 실행과정에서 발견될 수 있는 응용 프로그램의 구조적 문제점을 미리 점검하는 기능을 제공한다. 실행가능성 확인 스키마의 기술을 의해 필요한 PotentialUserOfSession 스키마는 다음과 같다.

PotentialUserOfSession

$$\begin{aligned}
&session?: \text{Session}\\
&potential_users!: USERS\\
&\hline
&potential_users = \{\ \forall\ u \in USERS \cdot\\
&\quad roles_of_session(session?) \subseteq roles_of_user(u)\ \}
\end{aligned}$$

PotentialUserOfSession 스키마는 입력으로 주어지는 세션을 수행할 수 있는 사용자들의 집합을 계산하는 기능을 수행한다. 수행결과로 주어지는 사용자 집합의 각 원소는 주어진 세션을 수행할 수 있는 역할들에 배정이 되어있음을 의미한다.

세션기반 임무분리 특성을 이용한 설계된 응용 프로그램의 실행가능성 확인 스키마의 정형적 기술은 다음과 같다.

FeasibilityCheck

MutexSessions

$\forall$ mutex_session $\in$ MutexSessions.mutex_sessions ·
($\forall s_i, s_j \in$ mutex_session · $s_i \neq s_j \wedge$
user_of_session(s_i) $\neq$ user_of_session(sj) $\wedge$
user_of_session(s_i) $\in$ PotentialUserOfSession(s_i) $\wedge$
user_of_session(s_j) $\in$ PotentialUserOfSession(s_j))

4.5 세션기반 응용 프로그램 명세 요소

응용 프로그램은 세션들과 세션들간의 제약조건의 집합으로 구성되지만 한편으로는 세션들의 연관성을 기반으로 세션들의 수행순서 결정과 제어기능을 담당한다. 표 1은 간단한 세션을 기반으로 한 응용 프로그램의 명세 요소(specification construct)를 표시하고 있다.

표 1 세션기반의 응용 프로그램 명세 요소

명세 요소	표 기 법
순차구조(sequential)	s_1 ; s_2
병렬구조(pararell)	s_1 || s_2
조건구조(conditional)	**if** <*condition*> **then** s_1 **else** s_2
반복구조(loop)	**while** <*condition*> **do** <*session-list*> **with max_loop** = <*integer*>
그룹(grouping)	(<*session-list*>)
종료(termination)	**abort**

표 1에서 <*session-list*>는 순차, 병렬, 조건, 그룹 요소에 의해 구성되는 하나 이상의 세션들로 구성되며, **abort** 요소는 응용 프로그램의 수행때 세션기반 임무분리 특성을 포함하는 제약조건이 만족되지 않을 때 응용 프로그램의 종료기능을 수행한다. 이 외에도 세션의 활동기간(lifetime), 세션간의 최대 이전시간(maximum transition interval) 등 응용 프로그램의 무결성에 영향을 미칠 수 있는 제약조건에 대한 명세 요소의 추가에 대한 연구가 필요하다.

5. 결 론

본 논문에서는 역할기반 접근제어 정책을 채택한 정보 시스템 환경에서 시스템이 관리하는 정보의 무결성 보장을 위한 세션기반 임무분리 특성을 제안하고, 제안된 특성을 기반으로 한 응용 프로그램 설계 및 운영 구조에 대하여 기술하였다. 또한 지금까지 진행된 역할기반 접근제어 정책에 관한 연구들에서 연구가 거의 진행되지 않은 세션의 정의와 특성들에 대하여 기술하였고, 세션을 기반으로 한 응용 프로그램 명세요소에 대한 연구도 제시하였다. 현재 역할기반 접근제어 정책은 상용 환경에서 정보 시스템의 보안 유지 및 접근제어 정책으로서 그 적합성이 우수하여 운영체제, 데이터베이스와 같은 시스템에서 채택, 사용되고 있어 본 논문에서 제안된 임무분리 특성과 무결성을 보장하는 응용 프로그램의 설계, 운영 구조가 자동화된 응용 프로그램 설계 및 실행가능성 점검 도구의 개발과 보다 안전한 시스템 구축에 활용될 수 있을 것으로 판단된다.

참고문헌

[1] David F. Ferraiolo, Janet A. Cugini, D. Richard Kuhn, "Role-Based Access Control(RBAC): Features and Motivations," *Proceedings of the 11th Annual Computer Security Applications Conferences*, December 1995, pp. 241-248.

[2] Virgil D. Gligor, Serban I. Gavrila, David Ferraiolo, "On the Formal Definition of Separation-of-Duty Policies and their Composition," *Proceedings of the IEEE Symposium on Security and Privacy*, May 1998, pp. 172-183.

[3] Ravi S. Sandhu, Edward J. Coyne, "Role-Based Access Control Models," *IEEE Computer*, February 1996, pp.38-47.

[4] Warwick Ford, *Computer Communications Security*, Prentice Hall, 1994

[5] U.S. Department of Defense, *Department of Defense Trusted Computer System Evaluation Criteria*, DOD 5200.28-STD, National Computer Security Center, December 1985.

[6] Ravi S. Sanhdu, Pierangela Samarati, "Access Control: Principle and Practice," *IEEE Computer*, September 1994, pp.40-48.

[7] John Barkley, "Comparing Simple Role Based Access Control Models and Access Control Lists," August, 1997.

[8] Ravi S. Sandhu, "Role Hierarchies and Constraints for Lattice-Based Access Controls," *Proceedings of the 4th European Symposium on Research in Computer Security*, September 1996.

[9] Ravi S. Sandhu, "Access Control: The Neglected Frontier," *Proceedings of the 1st Australian Conference on Information Security and Privacy*, June 1996.

[10] Ravi S. Sandhu, "Transaction Control Expressions for Separation of Duties," *Proceedings of the 4th Aerospace Computer Security Applications Conference*, December 1988, pp.282-286.

[11] , Ravi S. Sandhu, "Role-Based Access Control Features in Commercial Database Management Systems," *Proceedings of NISSC*, 1998.

[12] R. W. Baldwin, Naming and Grouping Privileges to Simplify Security

Management in Large Databases, *IEEE Symposium on Computer Security and Privacy*, 1990.

[13] K. R. Poland, M. J. Nash, Some Conundrums Concerning Separation of Duty, *IEEE Symposium on Computer Security and Privacy*, 1990.

[14] David Ferraiolo, Richard Kuhn, "Role-Based Access Controls," *Proceedings of the 15th NIST-NCSC National Computer Security Conference*, October 1978, pp.554-563.

[15] D. D. Clark, D. R. Wilson, "A Comparison of a Model for Computer Integrity," *Proceedings of the 1987 IEEE Symposium on Security and Privacy*, 1987, pp.184-194.

[16] S. Osborn, "Mandatory Access Control and Role-Based Access Control Revisited," *Proceedings of RBAC'97*, 1997, pp.31-40.

[17] Silvana Castano et al., *Database Security*, Addison-Wesley, 1994, pp18-34.

[18] J. M. Spivey, *Understanding Z*, Cambridge University Press, 1988.