通信情報保護學會論文誌
第 10 券 第 2 號, 2000. 6

작업간 비밀성을 보장하는 클래스 기반의 동적 의무분리 모델*

지 희 영**, 박 석***

Class-based Dynamic Separation of Duty Model for Ensuring Secrecy among Tasks

Hee-young Ji**, Seog Park***

요 약

기업 환경에서 정보의 무결성은 중요한 보안 요구사항이다. 의무분리 정책의 목적은 정보의 무결성을 필요로 하는 연산들을 여러 역할이나 사용자에게 분산시킴으로써 조직 내에서 관리하는 정보의 무결성 침해 가능성을 최소화하는 것이며 이것은 상업적 응용분야에서 중요하다. 전통적인 임의적 접근제어와 강제적 접근제어 정책의 대안인 역할 기반의 접근제어 기법은 응용에 따라 보호 객체들에 대한 접근을 역할들로 분류하여 단순한 권한 관리를 제공하며 의무분리 정책을 시행하기에 적합하다.

본 논문에서는 역할 기반의 접근제어에서 기존의 의무분리 안전성 조건의 취약성을 보완하여 응용 프로그램의 실행 단위인 클래스에 기반한 개선된 동적 의무분리 기법을 제안하고 상호 배타적인 부트랜잭션들을 포함하고 있는 중첩-트랜잭션을 대상으로 이를 적용한다. 또한 여러 작업들이 동시에 실행되는 환경에서 감염된 트로이언 목마에 의해 발생될 수 있는 정보의 유출 문제를 해결하고자 작업간 정보의 비밀성을 보장하는 동적 의무분리 모델을 제시한다. 제안한 모델은 기존의 무결성 보장을 위해 제시되었던 Clark-Wilson 모델과 다른 의무분리 모델에 비해 권한 관리가 용이하며, 동적으로 유지 관리되어야 하는 데이타의 양이 적고, 객체 접근 확인 절차가 단순하여 구현방법이 용이한 장점이 있다. 그리고 기존 의무분리 모델에서 고려되지 않은 병렬 수행 환경에서 작업 사이의 정보 유출 문제를 해결한다.

ABSTRACT

In commercial environments, an important security requirement is to ensure integrity of data. The purpose of separation of duty rules is to prevent fraud and error by dividing all operations which require integrity necessarily into several subparts or roles and requiring that each subpart be executed by a different person. Role-based access control(RBAC) is an alternative to traditional discretionary(DAC) and mandatory access control(MAC) policies, and is attracting increasing attention, particularly for commercial applications.

This policy simplifies management of authorization while providing an opportunity in specifying and enforcing separation of duty.

In this paper, we present a new method that is called 'Class-based Dynamic Separation of Duty' enhancing legacy safety condition for separation of duty in RBAC. Then we apply this method to nested transactions contained mutually exclusive subtransactions and design lattice based dynamic separation of duty model ensuring information secrecy among tasks in order to prevent Trojan Horse infected subjects from information leakage in concurrent environments. This model is better than other legacy integrity models such as Clark-Wilson or Sandhu model. It is simple to implement and has low overheads, both in the amount of secure information that must be maintained dynamically and in the efficiency of performing access control mediation. In addition, it could ensure information secrecy among tasks by designing lattice structure.

Keyword: *Database security, Role-based access control, Separation of duty*

* 본 연구는 정보통신연구진흥원 99년 대학 기초연구 지원비에 의한 결과임(과제 번호: C1-98-0744-00).
** 서강대학교 컴퓨터학과 데이터베이스 연구실 (jhyoung@dblab.sogang.ac.kr)
*** 서강대학교 컴퓨터학과 정교수 (spark@dblab.sogang.ac.kr)

I. 서 론

최근 컴퓨터 기술의 급격한 발전과 수많은 양의 정보로 인해 컴퓨터 보안 문제가 심각하게 대두되고 있다. 또한 컴퓨터 시스템과 정보 시스템들이 네트워크를 통해 서로 연결되고 정보 공유가 이루어짐에 따라 보안의 중요성은 더욱 커져가고 있다. 컴퓨터 보안의 주 목적은 정보의 보안, 무결성, 가용성을 보장하는데 있다.[1] 정보 보안의 주요 목적은 인가되지 않은 사람이 정보에 접근하거나 인가 없이 정보를 사용하는 것을 방지하는 것이다.[2] 본 논문에서는 데이타베이스 보안에 중점을 두고, 데이타베이스 내에 저장된 데이타에 대한 권한 없는 접근, 고의적인 파괴 및 변경으로 인한 무결성 침해 그리고 비일관성을 발생시키는 우발적인 사고로부터 데이타베이스를 보호하기 위한 방법론을 제시하고자 한다.

데이타베이스 보안을 위한 접근 방법으로는 군대 환경에 적합한 다단계 보안 데이타베이스 시스템과 상업 환경에 적합한 역할 기반의 데이타베이스 시스템을 들 수 있다. 다단계 보안 시스템은 각 시스템의 주체와 객체에게 보안 등급(security label)을 부여하고, 등급별로 분리된 정보를 유지하기 위해서 다중 보안 단계에서 정보를 처리하는 시스템이다. 즉, 사용자를 위한 하나 이상의 보안 인가 등급(security clearance level)과 시스템 내의 데이타들을 위한 하나 이상의 분류 등급(classification level)을 가진 시스템을 말하며 이러한 등급을 이용하여 인가 받지 않은 사용자로부터 기밀 정보의 유출을 제어하는 기법들을 제공한다. 한편, 역할 기반의 접근제어 시스템은 응용에 따라 보호 객체들에 대한 접근을 역할(role)들로 분류하여 데이타의 무결성을 보장하기 위해 정형 트랜잭션(Well-Formed Transaction)과 의무분리(Separation of Duty)의 원리로 정보를 처리하는 시스템이다.

역할 기반의 접근제어는 다양한 정책들을 구현할 수 있는 기법이다. 그 중 많은 상업적 응용들에서 중요한 요구 사항인 의무분리 정책은 무결성을 보장하기 위한 메커니즘으로 역할 기반의 접근제어 모델과 밀접히 관련되어 있다. 의무분리의 목적은 공모 혹은 사기행위를 막기 위해서 한 사람이 둘 이상의 부작업을 필요로 하는 하나의 작업에 모든 권한을 행하는 것을 막는 것이다. 이를 위해 몇가지 의무분리 정책들이 정의되어왔고,[3] 이들은 그 정책들이 시행되는 시점에 따라, 적용 대상 등에 따라 구분될 수 있다.[4] 현재까지 의무분리 정책과 관련된 연구는 주로 의무분리의 종류와 정형적인 기술을 중심으로 진행되었다.

본 논문의 구성은 다음과 같다. 2장에서는 역할 기반의 접근제어의 특성 및 모델의 소개와, Clark-Wilson, R. Sandhu 모델을 중심으로 기존의 의무분리 모델에 대해 살펴보고 기존 의무분리 모델에 관한 문제점을 제기한다. 3장에서는 클래스 기반의 개선된 동적 의무분리 기법을 제안하고 정보 시스템에서 무결성을 유지하면서 업무를 수행하는 단위인 트랜잭션을 대상으로 이를 적용한다. 4장에서는 여러 트랜잭션들이 동시에 실행되는 환경에서 의무분리 기법을 적용할 때 감염된 트로이언 목마에 의해 발생될 수 있는 클래스 사이에서 정보의 유출문제를 해결하기 위해 동적 의무분리를 위한 격자 구조 모델을 설계하고, 모델의 정확성을 증명한다. 5장에서는 기존의 관련 모델들과 제안한 모델을 비교, 분석하고 마지막으로 6장에서 결론 및 추후 연구 과제를 기술한다.

II. 관련 연구

2.1 역할 기반의 접근제어의 특성 및 모델

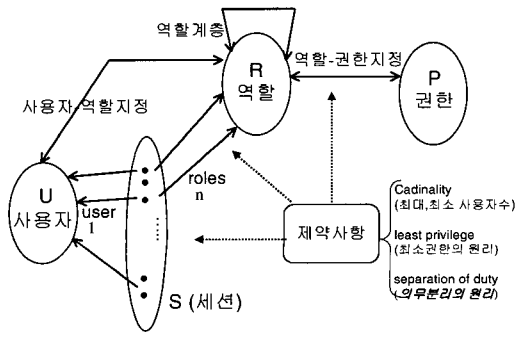
미국 국방성은 보안성 평가 기준과 보안정책 등을 규정한 TCSEC(Trusted Computer System Evaluation Criteria)을 1985년에 제정하였고,[5] 유럽에서도 정보보호 시스템의 평가 기준으로 ITSEC (Information Technology Security Evaluation Criteria)을 제정하였다. TCSEC에서는 강제적 접근제어(MAC:Mandatory Access Control)와 임의적 접근제어(DAC: Discretionary Access Control)의 두 가지 접근제어 정책에 대해 규정하고 있는데, 강제적 접근제어 정책은 군사환경이나 매우 제한적인 환경에서 제한된 수의 보안 관리자들에 의해 일정한 규칙에 따라 사용자의 정보에 대한 접근 권한을 통제한다. 반면에 임의적 접근제어 정책에서는 각 정보의 소유자들이 그들 임의의 판단으로 접근 권한을 다른 사용자에게 위임하거나 취소시킬 수 있다.

역할 기반의 접근제어(Role-Based Access Control)정책은 전통적인 임의적 접근제어(DAC) 그리고 강제적 접근제어(MAC) 정책의 대안으로 특히 상업적인 응용에서 관심이 증대되고 있다.[6] 역

할 기반의 접근제어(RBAC) 정책의 중요 사용 동기는 조직의 구조에 자연스럽게 사상(mapping)되는 기업에 특수한 보안정책을 명시하고 시행하기 위해 바람직하다는 점이다. 즉, 기업 환경에서는 기업마다 서로 다른 보안 요구 사항들과 정책들을 요구하기 때문에 강제적 접근제어나 자율적 접근제어 정책만으로는 이러한 요구 사항들을 만족시킬 수가 없다.[7,8] 일반적으로 기업환경에서는 강제적 접근제어 정책에서 중시하는 정보의 기밀성뿐만 아니라 정보의 무결성을 더 중요시하며 기업에서는 대체로 정보의 소유자가 임의적 접근제어와는 달리 기업 소속원이 아닌 몇 명의 관리자들에게 있는 특징을 가진다.

RBAC 정책에서 접근 결정은 개개인의 사용자들이 조직의 일부분으로서 가지는 역할에 근거한다. 접근 권한은 역할 이름에 의해 그룹화되고, 자원의 이용은 그러한 권한에 연관된 역할이 부여된 개개인들에게로 제약된다. 따라서 접근을 통제하기 위해 역할을 사용하는 것은 기업 특수의 보안 정책을 시행하고 개발하며 보안 관리를 능률화하기 위한 효율적인 도구가 된다. 즉, 정보에 대한 통제가 제한된 수의 관리자에 의해 수행되며 사용자가 대단히 많은 실제 기업 환경에 효율적으로 적용할 수 있기 때문에 현재 활발한 연구가 진행중이다.[9,10,11] 기본적인 RBAC 모델은 아래 그림 1과 같다.[7]

사용자에게 직접 객체에 대한 접근 권한을 부여하기보다는, 조직원으로서의 책임과 자격에 근거하여 역할에 소속원이 된다. 그리고 사용자에게 수행이 허용된 연산들은 그 사용자의 역할과 연관되어 있다. 따라서 조직상 기능들의 변경에 의해 새로운 연산이 설치되고, 오래된 연산들이 삭제될 때, 혹은 사용자의 역할에 대한 멤버쉽이 취소되거나 추가될 때 권한 관리와 유지보수가 단순화되는 장점이 있다. 이것은



〔그림 1〕 역할 기반의 접근제어 모델의 구성요소

역할(role), 역할 계층(role hierarchy), 관계성(relationship) 그리고 제약사항(constraints)들의 정의와 설계를 통해 사용자들의 활동을 정적으로 그리고 동적으로 통제하므로써 이루어지는 것이다. 이것은 관리자가 직접적으로 객체들 사이에 기초하여 접근제어 리스트(Access Control List)와 같이 낮은 레벨의 접근제어 기법을 시도하는 전통적이고 덜 직관적인 처리와는 다른 처리 과정이다.[12]

그림 1에서 역할은 부분 순서(partial order) '≥'를 가지고 조직화 되며 계층을 형성할 수 있다[1]. 세션(Session)은 한 사용자와 여러 개의 역할들의 집합으로 표현되는데, 사용자는 세션을 통해 자신에게 지정된 역할들 중 일부 또는 전체를 수행할 수 있다.[2] 제약사항은 모델의 각 구성 요소들과 매핑들에 적용되는 것으로 의무분리, 한 역할에 지정될 수 있는 최대, 최소 사용자의 수(cadinality), 선행 역할 그리고 최소권한의 원리(least privilege principle) 등이 될 수 있다.[3]

2.2 의무분리(Separation of Duty) 정책

의무분리(separation of duty) 정책은 정보의 무결성과 관련된 연산들을 여러 역할이나 사용자에게 분산시킴으로써 조직 내에서 관리하는 정보의 무결성 침해 가능성을 최소화하는데 목적이 있다. RBAC 관점에서 의무분리 특성은 표 1과 같이 세 가지 측면에서 형식적으로 묘사될 수 있다.[3]

정적 의무분리(Static Separation of Duty)는 사용자의 부정으로 인한 시스템의 무결성 침해 가능성을 막기 위해서 한 사용자는 상호 배타적인(mutually exclusive)두 개의 역할 둘 모두에 지정되어서는 안됨을, 반면에 동적 의무분리(Dynamic Separation of Duty)에서 사용자는 상호 배타적인 두 역할에 지정될 수 있지만, 실행시에 그 사용자를 대신하는 한 주체(subject)는 한 세션 안에서 두 개의 역할을 동시에 활성화시킬 수 없음을 명시한다. 마지막으로 연산상의 의무분리(Operational Separation of

1) 즉, 만약 x ≥ y 라면, x는 y의 조상으로 역할 x는 역할 y의 권한들을 모두 가질 수 있으며 x의 구성원들은 암시적으로 y의 구성원이 된다.

2) 이 때 세션에 의해 수행되는 역할들을 활성화된 역할(active role)이라고 한다.

3) 역할 기반의 접근제어(RBAC)는 다양한 정책들을 구현할 수 있는 기법으로 특히 의무분리 정책을 구현하기 위한 자연스런 메커니즘이다. 따라서 본 논문에서는 이 부분에 초점을 둔다.

〔표 1〕 의무분리 정책의 종류

의무분리 정책의 종류	형식적 표현
정적 의무분리	∀u∈user, ri,rj∈roles, ri≠rj : u∈role-members(ri) ∧ u∈role-members(rj) ⇒ ri∉mutually-exclusive-authorization(rj)
동적 의무분리	∀s∈subject, ri,rj∈roles, ri≠rj : ri∈active-roles(s)∧rj∈active-roles(s) ⇒ri∉mutually-exclusive-activation(rj)
연산상의 의무분리	∀s∈subject, r∈roles, f∈function : ¬(function-operations(f)⊆ $U_{r\in_{user\text{-}roles(u)}}$role-operations(r))

Duty)는 하나의 비즈니스 작업을 수행하기 위해 필요한 모든 연산들이 한 사용자가 갖는 역할들에 대한 연산들의 집합에 포함되어서는 안됨을 명시한다. 사업상의 중요한 작업이 공동작업을 포함하고 있다면, 사기 행위가 발생할 경우가 생길 수 있기 때문이다. 한편, 정적 의무분리는 동일 사용자를 상호 배타적인 역할에 지정하지 않음으로써 정보 시스템의 무결성을 유지하므로 결과적으로 많은 사용자가 필요하며, 시스템 운영의 부담이 커지고, 시스템 운영이 유동적이지 못하다. 반면, 동적 의무분리는 사용자가 비록 상호 배타적인 역할에 지정되더라도 실제 세션에 의해 활성화되는 역할들이 상호 배타적인 관계에 있지 않도록 유지하면 되므로 시스템 구성이 쉬워지고, 운영의 유연성이 커지는 장점이 있다.

2.3 기존의 의무분리 정책 모델

2.3.1 Clark-Wilson 모델

Clark과 Wilson은 의무분리를 목표로 어떤 한 사용자도 무결성이 중요시되는 작업 전체를 수행할 수 없도록 하기 위해서 트랜잭션과 권한들이 적절히 구조화 되어야 함을 암시하였고, 9개의 규칙에 기반하여 무결성 보장을 위한 프레임워크를 제시하였는데,[8] Clark-Wilson 무결성 모델의 기본적인 구성요소들로 제약 데이타 항목 CDI(constrained data item)와 변환 프로시져 TP(transformation procedure)를 명시했다. 그리고 임의적으로 조작될 수 있는 즉, 무결성 정책이 적용되지 않는 데이타 항목을 비제약 데이타 항목 UDI(unconstrained data item)로 명시하였다. 이 모델은 CDI에 대한 내적 일관성(internal consistency)을 보장하기 위한 프레임워크로 규칙 E1을, 외적 일관성(external consistency)인 의무분리 메커니즘을 제공하기 위해 규칙 E2를 아래와 같이 정의했다.

- **규칙 E1** : "시스템은 (Tpi, (CDIa, CDIb, CDIc, ...)) 유형의 릴레이션들의 리스트를 유지해야 한다. 즉, 각 TP에 대하여 TP가 참조하는 데이타 객체들의 관계를 나타낸다. 이 규칙에 따라 CDI의 조작은 단지 이러한 릴레이션에 명시된 TP에 의해서만 조작되어질 수 있다는 것을 보장해야 한다."
- **규칙 E2** : "시스템은 (사용자ID, TPi, (CDIa, CDIb, CDIc, ...)) 유형의 릴레이션들의 리스트를 유지해야 한다. 즉, 각 사용자에 대하여 TP 그리고 TP가 그 사용자를 대신해 참조하는 데이타 객체들의 관계를 나타낸다. 이 규칙에 따라 단지 이러한 릴레이션에 명시된 실행들만이 수행된다는 것을 보장해야 한다."

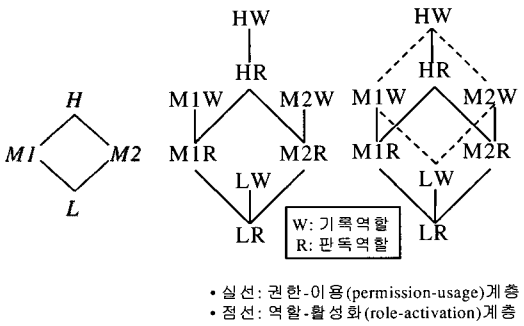
Clark과 Wilson은 위 규칙에서 명시한 릴레이션들이 명시적으로 유지되어야 함을 암시하고 있다. 그러나 TP들을 인증하고 사용자/TP의 트리플(triple)을 정의하는 것은 시스템 관리자에게 과중한 부담이 되며 위 규칙은 인증의 목적으로 언급된 것일 뿐 실제 구체적인 방법론을 제시하고 있지 않다.

2.3.2 Sandhu 모델

Ravi Sandhu는 RBAC 모델에서 역할 계층을 이용해 격자 기반의 접근제어 정책이 어떻게 시뮬레이트 되어질 수 있는가에 관하여 연구하였다.[13] 격자에서 판독, 기록 역할을 분리하여 각각에 대응되는 두 개의 등급을 정의하고 기록 역할은 판독 역할의 상위 역할로 어떤 조상도 가지지 않으며 계층이 존재하지 않는다[4](그림 2).

그림 2의 가장 왼쪽의 역할 계층에서 역할 M1와 M2가 서로 상호 배타적인 관계에 있는 역할 이라면 Richard Kuhn은 의무분리 요구사항을 만족하기 위해서 M1와 M2의 공통 상위 역할 H가 존재할 수 없다고 주장하였다.[4] 왜냐하면 H 역할의 활성화가 M1과 M2의 의무분리 관계를 위반하기 때문이다. 하지만, 동적 의무분리 정책이 정적 의무분리 정책과 다른점은 서로 다른 세션들에서는 M1와 M2를 활성화 시킬 수 있는 사용자가 존재한다는 데에

4) 즉, 기록 역할들은 모두 비교가능하지 않은 사적 역할(private role)로 존재한다.



〔그림 2〕 Sandhu의 동적 의무분리 모델

있다. 기존의 방법은 이러한 사용자들을 M1과 M2 두 역할 모두에 명시적으로 소속시킴으로써 해결하였지만 권한 관리를 단순화 시키는 RBAC의 기본적인 취지에 어긋난다. 따라서 Sandhu는 다음과 같은 제약사항으로 동적 의무분리 모델을 제시하였다.

- **제약 사항** : 사용자에게는 단지 기록 역할들만이 배정될 수 있으며 주어진 시간에는 단지 하나의 기록 역할이 활성화될 수 있다.

위의 제약사항에 의해 사용자는 한 세션에서 서로 상호 배타적 관계에 있는 역할들 중에서 하나의 기록 역할만을 활성화 시킬 수 있지만 다른 세션에서는 또 다른 기록 역할들을 활성화시킬 수 있다는 측면에서 동적 의무분리 요구사항을 만족시킨다. 즉, 기존의 계승적 역할 계층인 권한-이용 계층(permission-usage hierarchy)을 확장하여 역할-활성화 계층(role-activation hierarchy)을 분리해서 제시함으로써 권한-이용 계층에서는 가질 수 없는 공통 상위 역할이 존재할 수 있게 하였다.

하지만, Sandhu 모델의 한계는 첫째, 한 사용자가 한 세션에서 상호 배타적인 관계에 있는 역할들 중 하나의 기록 역할만을 활성화 시키는 방법 즉, 역할 기반으로 의무분리를 해결하기 때문에 상호 배타적인 부트랜잭션들이 동일 역할에 지정되어 있을 경우는 고려하지 않는다. 둘째, 하나의 역할에 대해 판독과 기록 역할을 각각 유지해야 하고 두 개의 역할 계층을 관리하는 오버헤드가 존재한다. 셋째, 여러 개의 작업이 동시에 수행되는 환경에서 작업간의 정보 유출문제에 대해 고려하지 않았다. 또한, 상호 배타적인 관계에 있지 않은 하위 기록 역할을 활성화 시킬 수 없기 때문에 병행성을 저하시킨다.

III. 트랜잭션 환경에서 클래스 기반의 동적 의무분리

3.1 연구의 동기

기존에 제시된 의무분리를 위한 안전성 조건(safety condition)은 정적 의무분리 시행에 초점을 둔 것으로 동일 사용자가 둘 이상의 부작업들로 구성된 한 작업에 대한 모든 권한의 부여를 막는 것이다. 의무분리를 위한 기존의 안전성 조건은 아래와 같이 표현된다.[4]

- **기존의 안전성 조건** (safety condition for separation of duty)

$$(\forall u \in USERS,\ \forall t \in TASKS,\ \forall i \in ROLES) : (C[t] \not\subset \cup_{u \in M[i]} P[i])$$

단, USERS는 사용자, TASKS는 여러 부작업들로 구성되어 있으며 의무분리가 요구되는 작업, ROLES는 역할집합, C[t]는 task -> $2^{privilege}$ 즉, 작업 task로부터 그러한 작업의 수행을 위해 필요한 권한들의 집합으로 사상(mapping)되는 함수, M[i]는 역할 i에 지정되어 있는 사용자들의 집합, P[i]는 역할 i에 지정되어 있는 권한들의 집합을 말한다.

그러나, 이는 여러 개의 부트랜잭션들로 구성되어 있는 중첩-트랜잭션에 기반한 동적 의무분리 특성에 적용하기에 불충분하며 이보다 유연성 있는 보다 엄격한 의무분리 특성에 관한 정의가 필요하다. 예를 들어, 하나의 수표가 다음 부작업들의 순서에 따라 준비되고, 발행되는 절차 T를 생각해 보자.

T={T1.1,T1.2,T1.3}

T1.1 : 한 점원이 수표를 준비한다.
T1.2 : 준비된 수표 발행이 한 감독관에 의해 승인된다.
T1.3 : 수표가 한 점원에 의해 발행된다.

점원과 감독관은 서로 상호 배타적인 관계에 있는 역할들이라고 가정한다면, 2.2절의 동적 의무분리 제약조건에 따라 실행시에 한 사용자에게 지정되어서는 안된다. 따라서 한 사용자 user1이 점원으로서 실행시에 1, 3번 부트랜잭션을 수행하고, 또 다른

사용자 user2가 감독관으로서 2번 부트랜잭션을 수행하는 경우를 생각해 볼 수 있다. 그러나 user1이 T를 수행하는 모든 권한을 갖지 않더라도 마찬가지로 임의로 수표를 준비, 발행할 수 있는 경우가 발생하게 된다. 즉, 비록 위의 동적 의무분리 조건과 의무분리를 위한 안전성 조건을 만족시킬 지라도 수표처리 과정의 무결성 침해가 발생할 가능성이 있는 것이다.

3.2 중첩-트랜잭션 설계와 운용구조

본 논문에서는 클래스에 기반한 동적 의무분리 시행을 제안하고 기업 정보 시스템에서 업무를 수행하는 단위인 트랜잭션을 대상으로 적용한다. 중첩-트랜잭션(nested-transaction)은 일반적으로 여러 개의 연관된 부트랜잭션(subtransaction)들과 제약조건들의 집합으로 구성되는데 제약조건은 부트랜잭션들 간의 무결성 침해 가능성, 수행되는 순서, 시간적인 제약 사항 등이 포함될 수 있다. 이렇게 중첩-트랜잭션을 부트랜잭션들의 집합으로 구조화시킴으로써 병렬성을 향상시킬 수 있고, 부트랜잭션이 시스템 결함으로 오류가 발생했더라도 그 영향을 지역화하여 처리할 수 있어 전체 트랜잭션을 철회(roll back)할 필요가 없다.

중첩-트랜잭션을 구성하고 있는 부트랜잭션들은 실제 실행시에 한 사용자에 대한 세션을 형성하게 된다. 여기서 이러한 메커니즘은 주어진 세션에 사용자를 지정하는 세션-사용자 지정과 주어진 세션에 역할들을 지정하는 세션-역할 지정의 두 부분으로 나누어 살펴볼 수가 있으며 세션-사용자 지정은 트랜잭션 실행시에, 세션-역할 지정은 트랜잭션 설계 과정시에 결정하는 것이 보다 개선된 유연성 있고 동작 성능을 높일 수 있는 시스템을 구성할 수 있다. 왜냐하면, 세션을 수행할 수 있는 사용자는 여러 명이고 트랜잭션을 실행할 때의 조건에 따라 그들 중 한명이 결정되는 방식으로 유동적인 반면에 각 세션을 구성하고 있는 부트랜잭션들을 수행하는 데에 필요한 권한들은 트랜잭션의 기능에 의해 대부분 미리 결정되어 있기 때문이다.

3.3 클래스 기반의 동적 의무분리 기법

본 논문에서는 RBAC에서 동적 의무분리 요구사항을 만족시키기 위해 응용 프로그램의 실질적 수행의 기본 단위인 여러 프로시져들로 구성된 클래스에 기반한 무결성을 유지 기법을 제안한다. 이를 위해서 프로그램 설계자는 한 사용자에게 동시에 지정되어서는 안되는 상호 배타적인 프로시져들을 결정한다. 따라서 개선된 동적 의무분리를 위한 안전성 조건은 공모, 사기를 통한 부정행위를 막기 위해 실행시에 동일 사용자에게 둘 이상의 상호 배타적인 관계에 있는 프로시져들에 대한 접근 권한을 가지는 것을 통제하는 것이다. 즉, 형식적으로 서술하면 아래와 같다.

- 개선된 클래스 기반의 동적 의무분리(Class-Based Dynamic Separation of Duty)

$\forall s \in subject,\ pi, pj \in procedures,\ pi \neq pj:$
$pi \in active\text{-}procedure(s) \wedge$
$pj \in active\text{-}procedure(s) \Rightarrow$
$pi \notin mutually\text{-}exclusive\text{-}activation(pj)$

클래스 기반의 동적 의무분리 기법은 상호 배타적인 부트랜잭션들을 반드시 다른 역할에 지정하지 않고서도 동적 의무분리 요구사항을 만족시킬 수 있는 유연성 면에서 장점에 있다. 즉, 역할의 수를 줄일 수 있고 따라서 역할 관리를 단순화시킬 수 있는 것이다. 다음의 예제를 살펴보자.

(예제 1) COI1: 수표 발행 절차
{T1.1점원,T1.2감독관,T1.3점원}
T1.1 : 한 점원이 수표를 준비한다.
T1.2 : 준비된 수표 발행이 한 감독관에 의해 승인된다.
T1.3 : 수표가 한 점원에 의해 발행된다.

(예제 2) COI2 : 물품 구매 절차
{T2.1점원 T2.2관리자 T2.3감독관}
T2.1 : 한 점원이 물품 구매 주문을 준비한다.
T2.2 : 한 구매 관리자에 의해 도착한 물품의 종류와 수량이 확인된다.
T2.3 : 감독관에 의해 구매된 물품에 대한 대금 지불이 승인된다.

물품의 실제 공급없이 주문과 지불이 만들어질 수 있는 사기 발생 가능성이 있기 때문에 위의 T2.1, T2.2, T2.3는 상호 배타적 관계에 있는 부트랜잭션

들이다. 각 COI 클래스에서 첫 줄은 중첩 트랜잭션을 구성하고 있는 부트랜잭션들과 역할, 수행 순서를 나타낸다. 역할 계층은 다음과 같다.

(단, 점원과 관리자는 상호 배타적 관계인 역할)

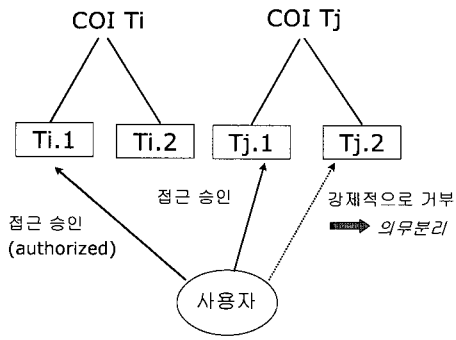
실제로 위의 (예제 1)을 살펴보면 은행에서 1,3번 부트랜잭션들은 점원이라는 동일 역할에 의해서 수행되는 것이 일반적이고 더 자연스럽다. 결국 예제의 1,2,3번 부트랜잭션들은 서로 상호 배타적인 프로시져들이므로 실행시에 서로 다른 사용자에 의해 수행되어야 한다.

IV. 병렬적으로 수행되는 트랜잭션 환경에의 적용

4.1 동적 의무분리 기법의 적용

하나의 트랜잭션이 아닌 여러 개의 트랜잭션이 동시에 병렬적으로 실행되는 상황을 생각해 보자. 이러한 환경에서 한 사용자는 여러 개의 작업에 배정되고, 여러 개의 부작업에 대해 권한이 부여된다. 이런 환경에 3.3절에서 제시한 클래스에 기반한 동적 의무분리 기법을 적용하고자 한다.

기업에서 수행되는 트랜잭션을 공용 정보(public information)에 대해 접근 권한이 부여된 공용 트랜잭션(public transaction)과 무결성이 요구되는 정보에 대해 접근 권한이 부여된 기업 트랜잭션(company transaction)으로 분류한다. 기업 트랜잭션은 서로 다른 상호 충돌(Conflict Of Interest) 중첩-트랜잭션들의 집합으로 구성되며, 각 중첩-트랜잭션은 여러 개의 부트랜잭션들로 구성되어 있다. 이 때 COI 중첩-트랜잭션이 하나 이상의 상호 배타적인 관계에 있는 부트랜잭션들을 포함하고 있을 때 그 트랜잭션은 의무분리 요구사항을 만족시켜야 한다. 즉, 한 사용자는 각 COI 중첩-트랜잭션에 대해서 상호 배타적 관계에 있는 부트랜잭션들 가운데 하나의 부트랜잭션에 대한 접근 권한을 가질 수 있게 한다. 예를 들어, 두 개의 COI 중첩-트랜잭션 Ti, Tj가 있고 각각은 상호 배타적인 두 개의 부트랜잭션을 포함하고 있다고 했을 때 그림 3. 의무분리를 만족하기 위해서는 동일 사용자는 각 COI 중첩-트랜



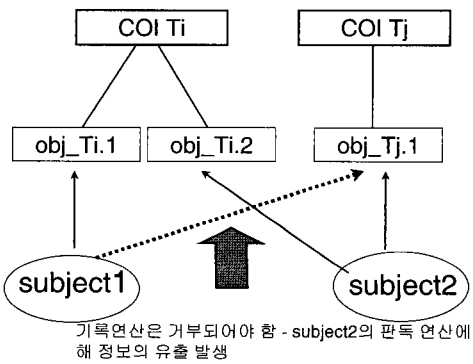
〔그림 3〕 COI 중첩-트랜잭션 상에서의 의무분리

잭션 Ti, Tj에서 두 개 이상의 부트랜잭션들에 대해 모든 접근 권한(여기서는 데이타에 대한 판독/기록 연산을 수행할 수 있는 권한이라 가정)을 가지는 것이 강제적으로 거부된다.

4.2 정보의 유출 문제

사용자 측면에서 의무분리 특성을 고려하면 실용적인 시스템에 실제로 적용하기에는 너무 제한적이므로 사용자, 세션, 주체(subject)를 구별하여 취급하므로써 동적 의무분리를 시행하려고 한다. 즉, 4.1절의 경우 한 사용자에게는 상호 배타적인 부트랜잭션 Tj.1과 Tj.2에 대한 권한이 배정될 수 있지만, 실제 사용자를 대신해 실행하는 주체에게는 둘 중 하나의 부트랜잭션에 대해서만 접근 가능하게 한다. 그러나 이 때 동적 의무분리 요구 사항을 만족하지만 클래스 간의 정보 유출 문제가 발생할 수 있다. 아래의 그림 4와 같은 상황을 보자.

사용자 1을 대신해 수행중인 주체 subject1은 Ti.1, Tj.1 객체들에 판독/기록을 허용하고, 사용자 2를 대신해 수행중인 주체 subject2에게는 Ti.2,



〔그림 4〕 정보의 유출 문제

Tj.1 객체들에 판독/기록을 허용하는 것은 클래스에 기반한 동적 의무분리 조건을 만족시킨다. 하지만, 사용자1의 권한을 갖는 감염된 트로이안 목마(Trojan Horse)나 바이러스가 Ti.1 객체에 관한 정보(판독 연산을 통해)를 Tj.1 객체로 전송(기록 연산을 통해)할 수가 있다. 이 때, subject1에 의한 정보의 유출로 인해 사용자2는 Ti.1, Ti.2 객체 둘 모두에 대한 정보에 대해서 판독 접근이 가능하므로 클래스 사이에 정보의 비밀성을 만족하지 못하는 경우가 발생한다. 즉, 정보의 유출이 발생한다. 따라서 subject1이 Tj.1 객체에 기록 연산을 위한 접근을 하지 못하게 해야 한다.

4.3 클래스들 사이에 비밀성을 보장하는 동적 의무분리 모델[5)]

4.3.1 기록 규칙

본 논문에서 제시하는 모델은 상업적(commercial) 환경의 기업 정보 시스템에서 작업간에 정보의 유출을 막아 비밀성을 보장하며, 권한 없는 사용자에 의한 데이타 기록을 막아 무결성 또한 보장하는데 목적이 있다. 이를 위해 모델은 시스템의 구성 요소인 주체와 객체에 지정하는 등급(Label)에 근거하여 주체가 객체에 접근할 수 있는가의 여부를 제어한다. 먼저 4.2절에서 언급한 정보의 유출 문제를 해결하여 비밀성을 보장하기 위해 기록 규칙을 정의한다. 모델에서 기업의 데이타 집합은 COI 트랜잭션 데이타 집합들로 분류되어 있다고 가정한다.

기록 규칙은 첫째, 두개 이상의 서로 다른 COI 트랜잭션 데이타 집합으로부터 객체들에 판독 연산을 수행했던 주체는 결코 기록 연산을 할 수 없다는 것과 둘째, 하나의COI 트랜잭션 데이타 집합으로부터 객체들에 판독 연산을 수행해왔던 주체는 바로 그 데이타 집합에만 단지 기록 연산을 할 수 있다는 것이다. 특히, 모든 주체가 하나의 COI 트랜잭션 데이타 집합에 판독과 기록을 하도록 제한하는 것은 수용할 수 있는 제약사항이고 의무분리 요구 사항에도 적합하다. 따라서 본 모델에서는 사용자 홍길동을 대신해 실행중인 어떠한 주체는 Ti 객체에 판독과 기록을 행하거나, 혹은 Tj 객체에 판독과 기록을 행하게 즉 둘 중 하나의 경우만을 허용한다. 그러나 사용자로서의 홍길동은 Tj.1과 Tj.2 객체들 둘 모두에 판독/기록을 허용하도록 지정할 수 있게 하므로써 동적으로 동작할 수 있다.

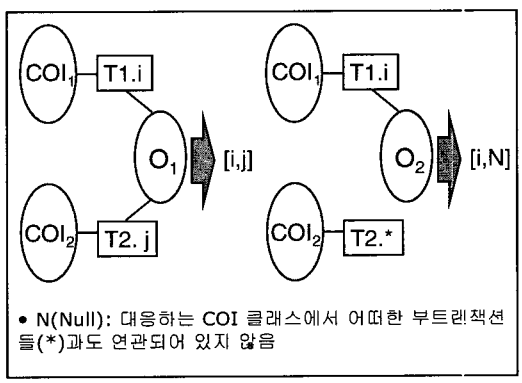
4.3.2 격자 구조의 설계

정보 흐름 정책(information flow policy)은 객체들이 격자 구조(lattice structure)로 등급이 부여될 것을 필요로 하는데 격자의 특징들은 일반적으로 합리적으로 수용된다.[14] 본 논문에서는 이러한 사실을 바탕으로 동적 의무분리 정책에서 정보의 흐름을 제어하기 위한 격자 구조를 설계한다. 우선 COI 중첩-트랜잭션 클래스와 부트랜잭션을 정의한다.

- **가정** : n개의 COI 중첩-트랜잭션 클래스가 존재하고 각 COIi는 mi 개의 상호 배타적인 부트랜잭션들을 포함한다.

$$COIS = \{ COI1, COI2, \ldots, COIn \},$$
$$COIi = \{ Ti.1, Ti.2, \ldots, Ti.mi \}, \ 1 \le i \le n$$

다음으로 시스템에서 기업내의 작업상 비밀 정보를 처리하는 부트랜잭션들에 근거하여 각각의 모든 객체에게 등급을 매긴다. 구체적으로 n개의 원소벡터 [i1, i2,..., in]로 객체 등급을 정의하려고 한다. 이 등급의 의미는 COI1 트랜잭션의 부트랜잭션 i1, COI2 트랜잭션의 부트랜잭션 i2 등과 관련된 정보를 갖고 있는 객체를 말한다. N(Null)의 의미는 대응하는 COI 트랜잭션 클래스에서 어떠한 부트랜잭션으로부터 정보를 가지고 있지 않은 객체를 의미한다. 즉, 모두가 N인 원소들을 갖는 등급은 공용 정보에 대응한다. 그림 5는 COI1, COI2 트랜잭션 클래스가 존재할 때의 예이다.



〔그림 5〕 제안한 모델에서 객체 등급의 예

5) 제안한 모델을 앞으로 클래스 기반의 DSD(Dynamic Separation of Duty) 모델이라고 칭하기로 한다.

이를 일반화하여 아래와 같이 정의한다.

• [클래스 기반의 DSD 모델의 등급 정의]

LABEL={{i1,i2,....,in} | i1∈*COI1, i2∈ * COI2 ,...., in∈*COIn}∪{SBL},
단, *COIi= COIi∪{N(null)}

또한 등급들 사이에 지배 관계(dominance relation)를 정의해야 한다. SL1[ik]는 등급 SL1의 k번째 원소를 나타낸다. 정의에 따라 모든 등급들은 모든 원소가 'N'인 시스템 최하위 등급을 지배하게 된다.

• [클래스 기반의 DSD 모델의 지배 관계 정의]
등급은 부분 순서화(partially ordered) 되어 지배(dominance) 관계에 근거하여 격자를 구성한다. 등급 SL1이 다른 등급 SL2를 지배한다는 것을 SL1≥SL2로 표시하고 두 등급은 다음과 같은 관계를 갖는다.

- SL1,SL2 ∈ LABEL, SL1≥SL2 ⇔ SL1[ik]=SL2[ik] or SL2[ik]=N,1≤k≤n
- 두 개의 등급 SL1과 SL2의 지배 관계가 SL1≥SL2 혹은 SL1≤SL2 둘 중 하나가 아니라면 두 등급은 비교할 수 없다.

격자 구조를 완성하기 위해서는 시스템에서 어떤 주체에게도 지정되지는 않지만, 시스템 최상위 등급을 나타내는 특별한 등급이 필요하므로 'SHIGH'라고 표기하기로 한다.

제안한 클래스 기반의 DSD 모델에서의 등급과 군사 보안을 위한 강제적 접근 제어(MAC)에서의 보안 등급을 비교해 보자. 우선, MAC의 대표적인 Bell-Lapadula 모델에서 기밀성 등급은 계층적 등급(hierarchical level)과 범주 집합(set of categories)으로 구성된다. 하지만 무결성 메커니즘이 MAC과 다른점은 첫째, 데이타 항목이 반드시 특수한 계층적 기밀성 레벨(hierarchical level)과 연관되어 있지 않다는 점이다. 오히려 데이타 항목을 조작하기 위해 허용된 프로그램의 집합과 연관되어 있다. 둘째, 상업적 보안 환경에서의 사용자에게는 특정 데이타 항목에 대해 판독, 기록연산을 수행하기 위한 권한이 부여되는 것이 아니라 특정 데이타 항목 위에서 특정 프로그램을 실행하기 위한 권한이 부여된다. 제안한 클래스 기반의 DSD 모델에서는 위의 사항들이 각각 객체, 주체 등급에 반영되어 있다. 다음으로 주체의 등급은 아래와 같은 보안 규칙에 따라 결정된다.

[보안 규칙 1] 사용자 등급 (UsEr Label)

UEL(user)={ul1,ul2,....,ulk,....,uln},
ulk={∀t∈COIk|role-trans(t)⊆ role-user(user)}, 1≤k≤n

단, 그러한 t가 존재하지 않는다면 ulk=N(Null). 즉, ulk는 사용자 user가 COIk 클래스에서 수행 가능한 역할에 지정된 부트랜잭션들의 리스트를 나타낸다.

[보안 규칙 2] 세션 등급 (SeSsion Label)

SSL(session)={sl1,sl2,....,slk,....,sln},
slk=ulk∪{N}, 1≤k≤n 중에 하나의 원소

- 활성화 가능 세션 집합(SSLS): 모든 가능한 slk(1≤k≤n)의 집합
- 각 COI에서 활성화 가능 세션 개수: NOSk(NumberOfSessionPerCOI)(user) =n(ulk) + 1, 1≤k≤n
- 사용자에 대한 모든 활성화 가능 세션의 개수: ANOS(AllNumberOfSessions)(user)= NOS1×NOS2...×NOSn

[보안 규칙 3] 주체 등급 (SuBject Label) :

SBL(subject)=SSLS 중에 하나의 원소

[보안 규칙 4] 등급 사이의 포함 관계 :

UEL⊇SSL⊇SBL

위에서 정의한 사항들을 바탕으로 등급의 집합 LABEL*=LABEL∪{SHIGH}과 지배 관계 '≥'을 가진 클래스들 사이에 비밀성을 보장하는 동적 의무분리 모델을 위한 격자 구조를 설계할 수 있다. 정보의 흐름은 지배 관계와는 반대방향이며, 반사,

이행, 대칭적인 성질을 만족한다. 모든 판독과 기록 연산은 주체에 의해 수행된다. 등급들 간의 지배 관계는 위에서 아래 방향으로 향하며, 이행과 반사적 관계를 나타내는 간선이 생략된 헤세 다이어그램(Hasse Diagram)으로 나타내진다.

다음으로 제안한 모델이 시행하는 보안 정책을 정의한다. 클래스 기반의 DSD 모델은 작업간 비밀성 보장을 위해 주체가 객체로의 접근을 시도할 때에, 아래에 있는 보안 정책을 만족할 경우에만 접근(판독, 기록연산만 고려)을 허용한다. 단, SL(S), SL(O)는 각각 주체와 객체의 등급이다.

[보안 정책 1]

주체 S는 다음의 조건을 만족하는 경우에만 객체 O에 대한 판독 연산을 수행할 수 있다.

조건 1: SL(S)≥SL(O)

[보안 정책 2]

주체 S는 다음의 조건을 만족하는 경우에만 객체 O에 대한 기록 연산을 수행할 수 있다.

조건 2: SL(S)≤SL(O)

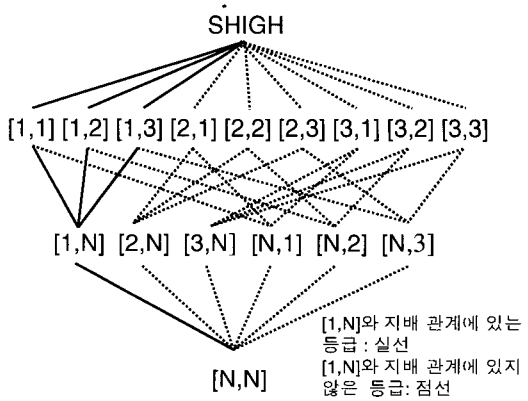
위와 같이 클래스 기반의 DSD 모델은 격자에서 하향 판독과 상향 기록을 시행한다.

4.4 제안한 모델에서 동적 의무분리 정책의 해석과 모델의 정확성 증명

위에서 제시한 모델이 의무분리 요구사항에 대하여 어떻게 해석될 수 있는지 예를 통해 살펴본다. 기업 데이타 집합이 두개의 중첩-트랜잭션 COI T1과 COI T2클래스로 구성되어 있고 각 COI는 세 개의 상호 배타적인 부트랜잭션들을 포함하고 있다고 가정하자(그림 6).

즉, COI T1={T1.1,T1.2,T1.3} COI T2={T2.1,T2.2,T2,3}로 표시할 수 있다.

만약 홍길동이 사용자로서 a라는 역할에 지정되었다면, 사용자로서의 등급은 [보안 규칙 1]에 의해서 [ul1,ul2] = {{T1.1,T1.2},{T2.2,T2.3}}와 같이 각 COI에 대한 트랜잭션 리스트로 구해진다. 그리고 활성화 가능한 세션 집합 SSLS은 [보안 규칙 2]에 의해 아래와 같이 구해진다.



[그림 6] 제안한 모델의 해석을 위한 예제

- **활성화 가능한 세션 집합 :**

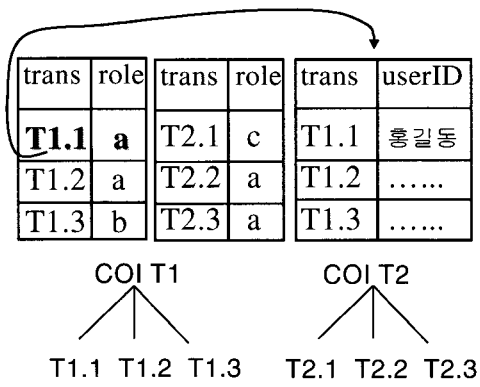
SSLS={ [N,N], [1,N], [2,N], [N,2], [N,3], [1,2], [1,3], [2,2], [2,3] }

클래스 기반의 동적 의무분리 기법 적용 ⇒
sl1: ul1={T1.1, T1.2, N} 중의 하나의 원소,
sl2: ul2={T2.2, T2.3, N} 중의 하나의 원소

즉, 홍길동.[N,N],..., 홍길동.[2,3]과 같은 9개(ANOS=9)의 세션들이 홍길동과 연관되게 되며 각각은 홍길동이 주어진 세션에서 로그인 하기를 원하는 등급에 대응한다. 이러한 선택 가능성들이 동적인 요구사항을 만족시킨다.

홍길동이 세션 [1,N]으로 로그인한다고 가정하자. 그러면 그 세션 동안에 생성된 모든 주체들은 [보안 규칙 3]에 의해 등급 [1,N]을 상속받는다. 즉, 주체의 등급은 그 주체를 생성하는 세션에 의해 결정된다([보안 규칙 4]). 그리고 모든 판독/기록 연산들은 단지 주체에 의해서 수행되고, 생성된 주체는 [보안 정책1]과 [보안 정책2]에 의해서 등급이 [N,N]인 공용 객체를 판독할 수 있고, 등급이 [1,N]인 객체를 판독/기록, 그리고 등급 [1,1], [1,2], [1,3] 그리고 SHIGH인 객체에 기록하는 것이 허용된다. 세션 내에서 홍길동을 대신해 실행하는 모든 주체의 등급은 [1,N]이므로 [2,*] (*={N,1,2})인 등급을 가진 어떤 객체와도 지배 관계가 성립하지 않기 때문에([클래스 기반의 DSD 모델의 지배 관계 정의]에 의해), 이러한 객체들에 대해서는 접근 권한을 가질 수 없다.

그림 7은 그림 6의 예제를 가지고 생성한 격자



[그림 7] [1, N] 등급을 갖는 객체의 지배 관계

구조를 보여준다. 시스템에서 모든 객체는 그림 7에 있는 등급들 중 하나로 부여된다. 공용 객체는 [N,N]이고, 하나의 부트랜잭션과 연관된 정보를 가진 객체는 [1,N],..., [N,3] 중의 하나로 등급이 부여된다. 또한, 두개의 부트랜잭션들로부터 관련 정보를 가진 객체는 [1,1],... ,[3,3]과 같이 등급화 된다. 결과적으로 홍길동은 사용자로서 T1.1과 T1.2 부트랜잭션을 수행할 수 있는 역할에 지정되었더라도 실제 실행시에는 부트랜잭션 T1.1과 연관된 객체들에 대해서만 판독과 기록 연산을 할 수 있고, 부트랜잭션 T1.1과 상호 배타적인 관계에 있는 부트랜잭션 T1.2와 연관된 객체에 대해서는 강제적으로 판독/기록 연산의 접근이 거부된다. 즉, 사용자로서 홍길동은 실행시에 로그인하려는 세션을 동적으로 선택할 수 있고, 위 격자 기반의 모델을 통하여 동적 의무분리 요구사항을 만족시키면서 또한 기록 규칙을 만족시킴으로써 감염된 트로이언 목마에 의해 발생할 수 있는 클래스들 사이의 정보 유출 문제를 해결할 수 있다.

다음으로 제안한 클래스 기반의 DSD모델의 정확성 즉, 동적 의무분리 정책을 보장함을 증명한다.

〈정리〉 : 제안한 모델은 동적 의무분리 정책을 보장한다.

〈증명〉 [정리]를 증명하려면 다음의 사항들을 증명해야 한다.

1) 시스템의 모든 주체 S는 각 COIk(1≤k≤n)를 구성하고 있는 상호 배타적인 부트랜잭션들을 한 세션 안에서 모두 활성화 시킬 수 없다. 즉, 개선된 동적 의무분리를 위한 안전성 조건을 만족시킨다.
2) 시스템의 모든 주체 S는 각 COIk(1≤k≤n)에 대해서 상호 배타적인 부트랜잭션들과 연관되어 있는 객체들을 한 세션 안에서 모두 접근(판독/기록 연산)할 수 없다.

[1] 사항에 대한 증명] :

제안한 모델의 [보안 규칙 2]에 의해 세션 등급(SSL)을 결정할 때에 클래스에 기반한 동적 의무분리 기법을 적용하므로, 세션 등급의 각 원소 slk(1≤k≤n)는 사용자 등급 UEL의 각 원소ulk (1≤k≤n) (사용자가 각 COIk에서 접근 가능한 트랜잭션들의 리스트)를 구성하고 있는 트랜잭션 집합 중에서 하나의 트랜잭션으로 제약된다. 즉, 세션 등급의 각 원소는 한 COI 클래스에서 하나의 트랜잭션으로 구성된다. 또한, [보안 규칙 3]에 의해 주체 등급은 활성화 가능한 세션 집합 SSLS 중에서 하나로 결정된다. 따라서, 주체 S는 한 COI를 구성하고 있는 상호 배타적인 부트랜잭션들 모두를 한 세션에서 활성화 시킬 수 없다.

[2] 사항에 대한 증명] :

COIk 트랜잭션 클래스는 상호 배타적인 관계에 있는 p, q 트랜잭션들을 포함하고 있으며 한 사용자 U는 두 부트랜잭션들에 대한 수행 권한이 있다고 가정하자.

[클래스 기반의 DSD 모델의 등급 정의]로부터 k번째 값(lk)이 각각 p, q이고 lk를 제외한 나머지 원소의 값은 동일한 두 개의 객체 등급 SLi과 SLj (i≠j)가 존재하고 두 객체의 등급은 아래와 같이 표현된다. SLi는 COIk 에서 p 부트랜잭션과 연관된 객체이며 SLj는 COIk 에서 q 부트랜잭션과 연관된 객체라는 의미를 갖는다. 다른 COIi(1≤i≤n, i≠k)에 대한 부트랜잭션들의 연관성은 같다고 하자.

SLi=[l1,l2,...,p,...,ln],
SLj=[l1,l2,...,q,...,ln]
⇒ SLi(lk)≠SLj(lk) ∧ SLi(lk)≠N ∧
SLj(lk)≠N ⇒ ¬(SLi SLj) ∧ ¬(SLi SLj)

[클래스 기반의 DSD 모델의 지배 관계 정의]에 의해 SLi와 SLj는 아무런 지배 관계가 존재하지

않는다. 그리고 가정과 [1) 사항에 대한 증명]에 의해 사용자 U를 대신해 실행하는 주체 S의 등급은 {s1,s2,...,p,...,sn} 혹은 {s1,s2,...,...p,...,sn} 중에 하나가 된다. 만약, si=li 혹은 si=N (1≤i≤n, i≠k) 값을 갖는다면 주체 S는 [보안 정책 1]과 [보안 정책 2]에 의해 SLi와 SLj 중 하나의 등급과 지배관계가 성립한다. 따라서 한 세션 안에서 등급이 SLi와 SLj인 객체 모두에 대해 판독, 기록 연산이 불가능하다. 즉, 둘 중 하나의 객체에만 접근할 수 있다. 따라서, 주체 S는 한 COI를 구성하고 있는 상호 배타적인 부트랜잭션들과 연관되어 있는 객체들을 한 세션 안에서 모두 접근(판독/기록 연산)할 수 없다.

[1) 사항에 대한 증명]과 [2) 사항에 대한 증명]에 의해서 제안한 모델은 동적 의무분리 정책을 만족한다.

V. 기존 모델과의 비교 및 분석

제안한 클래스 기반의 DSD 모델과 2장에서 언급한 기존의 Clark-Wilson 모델, Sandhu 모델을 비교 분석해 본다.

우선, Clark-Wilson 모델에서 동적 의무분리 정책을 구현하기 위해서는 2.3.1에서 언급된 것과 같이 완전한 규칙 E2가 필요할 뿐만 아니라 동적으로 갱신되는 릴레이션들의 집합을 유지해야 한다. 따라서 각각의 사용자에 기반하여 권한 확인을 하는 것은 실행시 유지되어야 하는 정보의 양에 있어서 큰 오버헤드이며 구현이 복잡해진다. RBAC에서는 개개의 사용자들에 대해 권한을 명세하는 릴레이션들을 유지하기 보다는 역할의 개념으로 대체하므로써 규칙 E2를 단순화시킬 수 있다. 또한 기업에서 사용자의 위치나 직급이 변경되면 그의 권한과 책임도 또한 변하게 되므로 제안한 클래스 기반의 DSD 모델에서처럼 명시적으로 정해져 있는 역할에 기반하여 권한 관리를 하는 것은 기업 환경에 더 자연스럽다. 한편, Clark과 Wilson은 무결성 보장을 위한 격자 모델에 대하여 "... 군사 환경과 기업 환경에서의 보안은 구별되어야 하며, 격자 모델은 기밀성 정책에는 적합하나 상업적 무결성 정책을 특성화하기에는 불충분하여 다른 메카니즘들이 필요하다." 라고 선언했다. 그러나 이것은 보편적으로 수용되지 않으며 본 논문에서 제시한 클래스 기반의 DSD 모델은 Clark-Wilson 모델에서 언급한 상업적 목적의 무결성 정책을 구현할 수 있는 격자에 기반한 메카니즘이다.

[표 2] 제안한 모델과 Clark-Wilson 모델에서의 객체 접근 확인 절차

Clark-Wilson 모델	제안한 클래스 기반의 DSD 모델
∀user∈USERS, tp∈TPS, cdi∈CDIS : CW_ACCESS(user,tp,(cdi1,..., cdin)) ⇒ Boolean	∀user∈USERS, trans∈COIk, 1≤k≤n : DSD_ACCESS(user,capable-user(trans)) ⇒ Boolean 단, capable-user(∀t∈COIk,1≤k≤n)={∀u ∈USERS:role-trans(t)⊆r ole-users(u)}

[표 3] 제안한 모델과 Clark-Wilson 모델의 메타데이타 비교

	클래스 기반의 DSD 모델	Clark-Wilson 모델
내적 일관성	객체 등급	(Tpi,(Cdi1,Cdi2,, Cdin)) 유형의 릴레이션
외적 일관성	(사용자ID,역할)과 (트랜잭션ID,역할)로 구성된 릴레이션	(사용자_ID,Tpi,(Cdi1,Cdi2,, Cdin)) 유형의 릴레이션
감사 로그 (audit log)	(트랜잭션_ID,사용자_ID)로 구성된 단순 로그	(사용자_ID,Tpi,(Cdi1,Cdi2,, Cdin)) 유형의 완전한 로그

클래스에 기반한 동적 의무분리 기법은 역할의 수를 줄여 관리를 단순화시킬 수 있는 장점을 제공하며 기업 시스템에서 총 트랜잭션의 수는 일반적으로 작기 때문에 각 사용자에 대해 트랜잭션 리스트를 유지하는 요구사항은 실제 합리적으로 수용될 수 있다. 또한 제안한 클래스 기반의 DSD 모델에서 접근제어 절차는 단순하다. 사용자가 로그인 하려고 하는 세션을 구성하고 있는 트랜잭션을 수행할 수 있는 역할에 그 사용자가 지정되었는지 확인하는 과정만 거치면 된다. 클래스 기반의 DSD 모델과 Clark-Wilson 모델에서의 객체 접근 확인 절차를 비교하면 표 2과 같다.

capable-user 함수는 입력으로 주어지는 트랜잭션 trans을 수행할 수 있는 사용자들의 집합을 생성하는 기능을 수행한다. 이 함수의 수행 결과로 얻어진 사용자는 해당 트랜잭션을 수행할 수 있는 역할에 지정되어 있음을 의미한다. 이 함수는 해당 트랜잭션 t를 수행하기 위해 필요한 역할 집합을 구하는 role-trans 함수와, 각 사용자 u에 지정되어 있는 역할 집합을 구하는 role-user 함수를 이용한다.

다음으로 필요한 메타데이타를 비교한다. 제안한 클래스 기반의 DSD 모델과 Clark-Wilson 모델에서 필요한 메타데이타를 비교한 것이 표 3이다.

〔표 4〕 제안한 모델과 기존 모델과의 비교

비교항목 \ 모델		제안한 클래스 기반의 DSD 모델	Clark-Wils on모델	Sandhu 모델
무결성 적용객체		COI 클래스 데이타 객체	제약 데이타 항목(CDI)	시스템 객체
무결성 비적용객체		공용정보 객체 (객체 등급: [N,N])	비제약 데이타 항목 (UDI)	구분하지 않음
의무분리 적용 대상		사용자-트랜잭션	사용자-TP-CDI	사용자-역할
작업간 비밀성		보장	보장안함	보장안함
무결성	내적 일관성	보장	보장	보장안함
	외적 일관성	보장	보장	보장
의무분리 방법론		클래스의 구조화 부분적 접근기록 이용(partial access history)	사용자-TP-CDI의 구조화 완전한 접근기록이용(full access history)	역할, 역할 계층의 분리, 상호 배타적인 역할관계 이용 유연성 저하(많은 역할 개수 필요)
트랜잭션의 병행성		보장	고려하지 않음	제약
객체 접근 확인 절차		역할에 근거, 단순	각 사용자에 근거한 릴레이션 리스트 확인, 복잡	역할에 근거
구현방법	권한 명세 데이타	사용자-단일 역할 지정 단일 역할 계층 유지	각 사용자에 근거한 사용자-TP-CDI 유형의 릴레이션 유지	사용자-기록역할 지정 두 개의 역할 계층 분리하여 유지
	동적 관리 데이타양	적음, 단순 감사 로그 기록	많음,완전한 전체 감사 로그 기록	고려하지 않음
	권한 관리 (변경, 추가)	적은 오버헤드 역할 관리 용이	큰 오버헤드 보안 관리자의 부담이 큼	Clark-Wilson 모델보다는 단순, 역할 분리에 따른 역할 관리를 위한 추가 오버헤드 필요

데이타에 대한 외적 일관성을 위해서 부트랜잭션들을 수행하는데 필요한 의무분리 요구사항을 명시하는 메타데이타가 필요하며 실제 실행을 기록하기 위해 감사 로그가 요구된다. 이를 통해 제시한 모델은 시스템에서 정보 객체는 단지 권한이 부여된 트랜잭션들에 의해서만 수정될 수 있다는 규칙[6]을 만족하며, 트랜잭션의 실행은 객체를 유효한 상태(valid state)에서 또 다른 유효한 상태로 변환시킨다.[7] 다만, 역할에 기반하여 접근 제어를 하므로 역할을 관리하기 위해 유지해야 하는 추가 엔터티가 필요하지만 이들은 모두 고정되어 있고(fixed) 이를 통해 권한 관리를 효율적으로 할 수 있는 장점이 있다.

다음으로 제안한 클래스 기반의 DSD 모델과 Sandhu 모델을 비교, 분석해 본다. 제안한 모델에서는 3.3절에서 제시한 (예제 1)과 (예제 2) 모두를 취급하며, 단일 역할 계층을 사용하므로 역할의 수를 줄여 관리를 단순화 시킬 수 있다. Sandhu 모델에서는 (예제 1)과 같은 상황은 다룰 수 없다. 한편, (예제 2)와 같은 경우에 감독관은 한 세션에서 역할-활성화 계층을 이용하여 직원, 점원, 관리자, 감독관의 기록 역할들 중 하나의 역할을 활성화하여 작업을 수행함으로써 동적 의무분리를 만족할 수 있지만 한 세션에는 하나의 기록 역할만을 활성화할 수 있다는 제약 사항때문에 상호 배타적 관계에 있지 않은 직원으로서의 기록 작업을 할 수 없는 단점이 있고 작업들 사이에 발생할 수 있는 정보의 유출 문제를 해결하지 못한다. 위의 사항들은 아래 표 4로 요약될 수 있다.

6) Clark-Wilson모델에서 외적 일관성에 해당한다.
7) Clark-Wilson모델에서 내적 일관성에 해당한다. 또한 객체의 유효한 상태는 무결성이 보장된 상태를 의미한다.

VI. 결론 및 추후 연구 과제

본 논문에서는 전통적인 임의적 접근제어와 강제적 접근제어 정책의 대안으로 특히 기업 환경에서 관심이 증가되고 있는 역할 기반의 접근제어에서 데이타베이스 내에 저장된 데이타에 대한 권한 없는 접근, 고의적인 파괴 및 변경을 발생시키는 우발적인 사고로부터 데이타베이스를 보호하여 무결성을 보장하기 위해 기존의 의무분리 안전성 조건의 취약성을 보완한 클래스 기반의 개선된 동적 의무분리 기법을 제안하였다.

그리고 제안된 기법을 상호 배타적인 관계에 있는 부트랜잭션들을 포함하고 있는 중첩-트랜잭션에 적용하여 동적 의무분리 요구사항을 만족시키기 위해서 주체, 세션 기반에서 새롭게 해석하였다. 이 기법은 시스템 운영의 유연성을 향상시키고 역할 관리를 단순화시킬 수 있는 장점을 가진다. 또한 여러 작업들이 동시에 실행되는 환경에서 작업간 비밀성을 위한 격자 구조에 기반을 둔 동적 의무분리 모델을 제시하여 감염된 트로이언 목마에 의해 발생될 수 있는 정보의 유출 문제를 보안규칙을 통하여 정보의 흐름을 한 방향으로 제어가 가능하게 하므로써 해결하였다. 제안한 모델은 무결성 보장을 위해 제시되었던 기존 모델에 비해 권한 관리가 단순하며 동적으로 유지, 관리되어야 하는 데이타의 양이 적고, 객체 접근 확인 절차가 단순하여 구현방법이 용이한 장점이 있다. 또한 기존의 의무분리 모델에서 고려되지 않은 작업간 정보의 비밀성 또한 보장한다.

응용 프로그램의 실질적 수행의 기본 단위인 클래스는 여러 프로시져들과, 프로시져들 간의 연관성을 기반으로 프로시져들의 수행 순서 결정과 제어 기능을 담당하는 제약 조건의 집합으로 구성된다. 응용 프로그램은 순차(sequential), 병렬(parallel), 조건(conditional), 반복(loop), 그룹(grouping) 등의 다양한 여러 구조를 가질 수 있으므로 이들 구조를 표현하는 형식적인 명세 요소를 필요로 한다. 따라서 제안한 클래스 기반의 동적 의무분리 기법을 적용할 대상인 클래스에 대하여 무결성에 영향을 미칠 수 있는 제약 조건의 명세 요소들에 대한 추가 연구가 앞으로 더 필요하다.

참 고 문 헌

[1] Ravi Sandhu and Sushil jajodia, "Integrity mechanism in database management systems", *Proceedings of the 13th NIST-NCSC National Computer Security Conference*, 10, 1990.

[2] Ravi Sandhu, "Transaction Control Expressions For Separation Of Duties", *Proceedings of the 4th Aerospace Computer Security Applications Conference*, December 1988.

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

[4] Richard Kuhn, "Mutual Exclusion of Roles as a Means of Implementing Separation of Duty in Role-Based Access Control Systems", *Second ACM Workshop on Role-Based Access Control*, 1997.

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

[6] David Ferraiolo and Richard Kuhn, 'Role-Based Access Control", *Proceedings of the15th National Computer Security Conference*, 1992.

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

[8] David D. Clark and David R. Wilson, "A Comparison of Commercial and Military Computer Security Policies", *Proceedings of the 1987 IEEE Symposium on Security and Privacy*, pp. 184~194, 1987.

[9] Ravi Sandhu and Hal Feinstein, "A Three Tier Architecture for Role-based Access Control", *Proceedings of the 17th NIST-NCSC National Computer Security Conference*, 10, 1994.

[10] Barkley, Cincotta, Ferraiolo, Gavrilla and Kuhn, "Role Based Access Control for the World Wide Web", *20th National Computer Security Conference*, 1997.

[11] Chandramouli Ramaswamy and Ravi Sandhu, "Role-Based Access Control Features in Commercial Database Management Systems", *NISSC the National Information Systems Security Conference*, 1998.

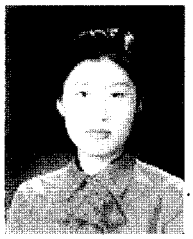
[12] John Barkley, "Comparing Simple Role Based Access Control Models and Access Control Lists", *Second ACM Workshop on Role-Based Access Control*, 1997.

[13] Ravi Sandhu, "Role Activation hierarchies", *Proceedings of the 3rd ACM Workshop on Role-Based Access Control*, Fairfax, Virginia, October 1998.

[14] Ravi Sandhu, "A lattice interpretation of the chinese wall policy", *Proceedings of 15th NIST-NCSC National Computer Security Conference*, Baltimore, pp. 221~235, October 1992.

〈著者紹介〉

박 석 (Seog Park) 정회원
1978년 2월 : 서울대학교 계산통계학 학사
1980년 2월 : 한국과학기술원 전산학 석사
1983년 8월 : 한국과학기술원 전산학 박사
1983년 9월 ~현재 : 서강대학교 컴퓨터학과 정교수
〈관심분야〉 실시간 데이터베이스, 데이터베이스 보안, 웹 데이타베이스



지 희 영 (Hee-young Ji) 정회원
1997년 2월 : 서강대학교 전자계산학과 학사
2000년 2월 : 서강대학교 컴퓨터학과 석사
〈관심분야〉 데이터베이스 보안, 실시간 데이터베이스, 트랜잭션 병행수행 제어 기법