역할 기반의 접근제어 시스템에서 동적 의무분리 만족을 위한 설계 방법

지 희 영*, 박 석
서강대학교 컴퓨터학과 데이터베이스 연구실

A Design for Dynamic Separation of Duty in Role-Based Access Control Systems

Hee-young Ji and Seog Park
Database Research Lab , Dept of Computer Science, Sogang University
{jhyoung, spark}@dblab sogang ac kr

요 약

역할 기반의 접근제어 시스템은 응용에 따라 보호 객체들에 대한 접근을 역할들로 분류하여 정당 트랜잭션과 데이터의 무결성을 보장하는 의무분리의 원리로 정보를 처리하는 시스템으로 편리하고 단순한 권한 관리를 제공한다 본 논문에서는 기업 환경에 적합한 역할 기반의 접근제어 시스템에서 데이터베이스 내에 저장된 데이터에 대한 권한 없는 접근, 고의적인 파괴 및 변경을 야기하는 사고로부터 데이터베이스를 보호하기 위해 정보의 무결성 보장을 위한 의무분리의 원리를 제안한다 그리고 제안된 원리를 기반으로 하여 지용 디성으로 상호 배타적인 부트랜잭션들을 포함하고 있는 중첩-트랜잭션을 생각해 보았으며, 여기에 동적 의무분리 요구사항을 만족시키기 위해서 주체, 세션 기반에시 세롭게 해석하였다 이 기법은 시스템 운영의 유연성을 향상시키고, 역할 관리를 단순화시키는 장점을 가진다. 또한 여러 트랜잭션들이 동시에 실행되는 환경하에 정보의 무결성 유지를 위해서 본 논문에서 제안한 의무분리 기법을 적용하였다. 이때 감염된 트로이인 목마에 의해 발생될 수 있는 정보의 유출문제를 해결하고자 의무분리를 위한 격자 구조를 설계하고 이를 바탕으로 해석하였다.

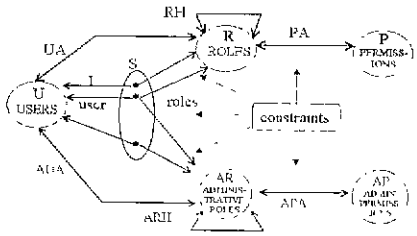
1. 서론

최근 컴퓨터 기술의 급격한 발전과 수많은 양의 정보로 인해 컴퓨터 보안 문제가 심각하게 대두되고 있다 본 논문에서는 데이터베이스 보안에 중점을 두고, 데이터베이스 내에 저장된 데이터에 대한 권한 없는 접근, 고의적인 파괴 및 변경으로 인한 무결성 침해 그리고 비일관성을 발생시키는 우발적인 사고로부터 데이터베이스를 보호하기 위한 방법론을 제시하고자 한다 역할 기반의 접근제어 시스템은 응용에 따라 보호 객체들에 대한 접근을 역할(role)들로 분류하여 정당 트랜잭션(Well-Formed Transaction)과 데이터의 무결성을 보장하는 의무분리(Separation of Duty)의 원리로 정보를 처리하는 시스템이다 역할 기반의 접근제어는 다양한 정책들을 구현할 수 있는 기법으로 많은 상업적 응용들에서 증요한 요구사항인 의무분리 정책을 구현하기 위한 자연스런 메커니즘이다 현재까지 의무분리 정책과 관련된 연구는 주로 의무분리의 종류와 관계, 정형적인 기술을 중심으로 진행되었다 [2,6] 이 논문에서는 기업 환경에서 역할 기반의 접근제어 시스템에서, 실제 응용 프로그램의 실행 단위인 세션에 기반한 동적 의무분리 시행 기법을 제안하고 트랜잭션(nested-transaction)을 대상으로 이를 적용한다 또한 여러 트랜잭션들이 동시에 실행되는 환경에서 제안된 의무분리 기법을 적용하고 이때 발생될 수 있는 정보의 유출문제를 해결하고자 의무분리를 위한 격자 구조를 설계하고 이에 기반하여 해석한다

2. 역할 기반의 접근제어 정책

2.1 역할 기반의 접근제어 정책의 특성 및 모델의 구성요소

미국 국방성은 보안성 평가 기준과 보안정책 등을 규정한 TCSEC(Trusted Computer System Evaluation Criteria)을 1985 년에 제정하였고, 강제적 접근제어(MAC Mandatory Access Control)와 임의적 접근제어(DAC Discretionary Access Control)의 두가지 접근제어 정책에 대해 규정하고 있다 그러나 기업 환경에서는 기업마다 서로 다른 보안 요구사항들과 정책들을 요구하기 때문에 강제적, 자율적 접근제어 정책만으로는 부적합하다고 판명되어 왔다[4]. 역할에 기반한 접근제어 (Role-Based Access Control) 정책은 전통적인 이러한 정책들(MAC,DAC)의 대안으로 특히 상업적인 응용에서 관심이 증대되고 있다[1] 이 기법의 중요한 사용 동기는 조직의 구조에 자연스럽게 메핑시키는 기업에 특수한 보안정책을 명시하고 시행하기 위해 바람직하다는 점이다 접근 결정은 개개인의 사용자들이 조직의 일부분으로서 가지는 역할에 근거한다 접근 권한은 역할 이름에 의해 그룹화되고, 정보의 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. 이러한 역할 기반의 접근제어 정책은 정보에 대한 통제가 제한된 수의 관리자에 의해 수행되며 사용자가 대단히 많은 실제 기업 환경에 효율적으로 적용할 수 있기 때문에 이에 대해 활발한 연구가 현재 진행중이다 역할 기반의 접근제어 모델과 관리상 모델, 그리고 이에 대한 정형적 기술, 역할 기반의 접근제어 시스템 구조와 설계, 구현 방법에 대한 연구가 이루어지고 있으며, 분산 환경 혹은 웹 환경에서 역할 기반의 접근제어 기법을 적용하는 연구가 이루어지고 있다 역할 기반의 접근제어 모델의 구성요소는 아래 [그림 1]과 같다[5]



[그림 1] 역할 기반의 접근제어 모델의 구성요소 (UA 사용자-역할 지정 RH 역할계층, PA 역할-권한 지정, S(Session) 세션, AUA 관리자-역할 지정, ARH 관리역할 계층)

사용자(users)는 사람 혹은 자동화된 에이전트를 말하며, 역할(roles)은 조직 내에서 일의 기능 혹은 일의 타이틀이다 권한(permissions)은 시스템 안에서 하나 이상의 객체들에 대한 특별한 접근 모드의 승인 혹은 연산을 수행하기 위한 및및 권한들을 나타낸다 역할은 부분적인 순서(partial order) '≥' 내에서 역할 계층으로 조직화될 수 있다 세션(Session)은 한 사용자와 여러 개의 역할들의 집합으로 표현되는데, 사용자는 세션을 통해 자신에게 지정된 역할들 중 일부 또는 전체를 수행할 수 있으며 이 때 세션에 의해 수행되는 역할들을 활성화된 역할(active role)이라고 한다 제약사항(constraint)은 위에서 정의된 역할 기반의 접근제어 모델의 각 구성 요소들과 메핑들에 적용될 수 있는 것으로 의무분리(Separation of Duty), 한 역할에 지정될 수 있는 최대 사용자의 수(cadinality), 그리고 사용자가 특정 일을 수행함에 있어 필수 권한만을 부여하는 최소권한의 원리(least privilege principle) 등이 될 수 있다

2.2 의무분리(Separation of Duty) 정책

의무분리 정책은 공모 혹은 사기 행위를 막기 위해서 정보의 무결성과 관련된 연산들을 여러 역할이나 사용자에게 분산시킴으로써 조직 내에서 관리하는 정보의 무결성 침해 가능성을 최소화하는데 목적이

있다 역할 기반의 접근제어 모델에서 의무분리의 특성은 세기지 측면에서 형식적으로 묘사할 수 있다[2] 사용자의 부정으로인한 시스템의 무결성 침해 가능성을 막기 위해서 한 사용자는 상호 배타적인(mutually exclusive)두 개의 역할 둘 모두에 지정되어서는 안됨을 명시하는 정적 의무분리(Static Separation of Duty)와 사용자는 상호 배타적인 두 역할에 지정될 수 있지만, 실행시에 그 사용자를 대신하는 한 주체(subject)는 한 세션 안에서 상호 배타적인 두 개의 역할을 동시에 활성화시킬 수 없음을 명시하는 동적 의무분리(Dynamic Separation of Duty). 사업상 하나의 작업(business function)을 수행하기 위해 필요한 모든 연산들이 한 사용자가 갖는 역할들에 대한 연산들의 집합에 포함되어서는 안됨을 명시하는 연산상의 의무분리(Operational Separation of Duty)가 그것이다 정적 의무분리는 많은 사용자가 필요하며 시스템 운영의 부담이 커지고 시스템 운영이 유동적이지 못한 단점을 가진 반면에, 동적 의무분리는 시스템 구성이 쉬워지고 운영의 유연성이 커지는 장점이 있다 대신, 활성화될 역할들간의 관계를 확인하는 과정이 추가적으로 필요하다

3. 중첩-트랜잭션에서 세션에 기반한 동적 의무분리

이 논문에서는 정보 시스템에서 무결성을 유지하면서 업무를 수행하는 단위인 트랜잭션(transaction)을 대상으로 동적 의무분리 정책을 적용하려 한다 중첩-트랜잭션(nested-transaction)은 일반적으로 여러 개의 연관된 부트랜잭션(subtransaction)들과 제약조건들의 집합으로 구성된다 또한 트랜잭션들은 미리 정의된 역할들에 의해서만 수행되며 고유 식별자를 가지고 있다. 중첩-트랜잭션은 이렇게 하나의 트랜잭션을 부트랜잭션들의 집합으로 구조화시킴으로써 병렬성을 향상시킬 수 있고, 부트랜잭션이 시스템 파괴(crash) 등의 이유로 결함(failure)이 발생했더라도 전체 트랜잭션을 철회(roll back)할 필요가 없는 장점이 있다. 중첩-트랜잭션을 구성하고 있는 부트랜잭션들은 실제 실행시에 한 사용자에 대한 세션을 형성하게 된다 기존에 제시된 의무분리를 위한 안전성 조건(safety condition)은 C[t] task -> $2^{privilege}$ 를 둘 이상의 부분 작업을 포함하면서 의무분리를 필요로 하는 작업 task 로부터 그러한 작업의 수행을 위해 필요한 권한들의 집합으로 매핑되는 함수라고 할 때, 어떠한 한 사용자가 한 작업 t의 모든 권한을 가질 수 없음을 보장하기 위해 C[t] 만의 모든 권한들에 대한 접근을 갖지 않도록 통제하는 것이다[6] 그러나, 여러 개의 부트랜잭션들로 구성되어 있는 중첩-트랜잭션에서 동적 의무분리 정책을 적용할 때 위의 조건만으로는 불충분하며 이보다 엄격한 의무분리 특성에 관한 정의가 필요하다 세 개의 부트랜잭션들로 구성되어 수표가 준비되고, 발행되는 중첩-트랜잭션의 예를 살펴보자

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

동적 의무분리 조건에 따라 한 사용자 user1 이 점원으로서 실행시에 1, 3 번 부트랜잭션을 수행하고, 또 다른 사용자 user2 가 감독관으로서 2 번 부트랜잭션을 수행할 경우(점원과 감독관은 서로 상호 배타적인 관계에 있는 역할이므로)에 user1 이 임의로 수표를 준비, 발행할 수 있는 경우가 발생하게 된다. 즉, 비록 위의 동적 의무분리 조건과 안전성 조건을 만족시킬 지라도 수표처리 과정에서 무결성 침해가 발생할 가능성이 있는 것이다 따라서 이 논문에서는 역할 기반의 접근제어 시스템에서 동적 의무분리 요구사항을 만족시키기 위해 실질적 수행의 기본 단위인 세션에 대한 무결성을 유지 기법을 제안한다 이 방법은 한 사용자에게 동시에 지정되어서는 안되는 상호 배타적인 세션들을 결정해서 세션-사용자 지정시에 제약사항으로 적용한다 즉, 실제 실행시에 동일 사용자에게 둘 이상의 상호 배타적인 관계에 있는 부트랜잭션에 대한 접근 권한을 가지는 것을 통제하는 하는 것이다.

- **세션에 기반한 동적 의무분리(Dynamic Separation of Duty):**

  ∀ s :subject, si,sj:sessions : si ≠ sj :

  si ∈ active-sessions(s) ∧ sj ∈ active-sessions(s)

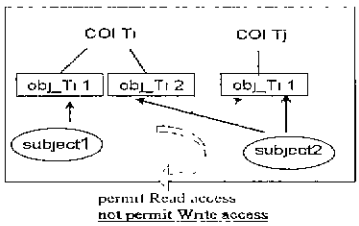
  ⇒ si ≠ *mutually-exclusive-activation* (sj)

이 기법은 위의 수표발행 절차 트랜잭션의 예처럼 상호 배타적인 부트랜잭션들을 반드시 다른 역할에 지정하지 않고서도 동적 의무분리 제약사항을 만족시킬 수 있기 때문에 역할의 수를 줄일 수 있고 따라서 역할 관리를 단순화시킬 수 있는 장점이 있다 위의 예에서는 결국 1,2,3 번 부트랜잭션은 실행시에 서로 상호 배타적인 세션들이므로 위의 원리에 따라 서로 다른 사용자에 의해 수행된다

4. 병렬적으로 수행되는 트랜잭션 환경에의 적용

4.1 동적 의무분리 정책의 적용과 정보 유출문제

하나의 트랜잭션이 아닌 여러 개의 트랜잭션이 동시에 실행되는 환경에서는 한 사용자에게 여러 개의 작업이 배정되고, 여러 개의 부작업에 대해 권한이 부여된다 이런 환경에 위에서 언급한 세션에 기반한 동적 의무분리 정책을 적용하고 이 때 발생할 수 있는 정보의 유출문제(information disclosure)에 대한 해결책을 제시하고자 한다 기업에서 수행되는 트랜잭션은 공용 정보(public information)에 대해 접근 권한이 부여된 공용 트랜잭션(public transaction)과 기업 기밀 정보에 대해 접근 권한이 부여된 기업 트랜잭션(company transaction)으로 분류될 수 있다 기업 트랜잭션은 서로 다른 상호 충돌 COI(conflict of interest) 중첩-트랜잭션들의 집합으로 구성되며, 각 중첩-트랜잭션은 여러 개의 부트랜잭션들로 구성되어 있다고 하자 이때 COI 중첩-트랜잭션이 하나 이상의 상호 배타적인 관계에 있는 부트랜잭션들을 포함하고 있을 때 그 트랜잭션은 의무분리 요구사항을 만족시켜야 한다 이를 위해서 본 논문에서는 실행시 한 사용자는 각 COI 중첩-트랜잭션에 대해서 상호 배타적 관계에 있는 부트랜잭션들 중 동시에 하나의 부트랜잭션에 대한 접근 권한(데이터에 대한 판독/기록 연산을 수행할 수 있는 권한이라 가정)만을 가질 수 있도록 규제할 것을 제안한다 예를 들어, 두 개의 COI 중첩-트랜잭션 Ti, Tj 가 있고 각각은 상호 배타적인 관계에 있는 두 개의 부트랜잭션 Ti 1, Ti 2 과 Tj 1, Tj.2 가 있다고 하자. 한 사용자에게는 상호 배타적인 부트랜잭션 Tj 1 과 Tj.2 에 대한 접근 권한이 배정될 수 있지만, 실제 실행시 그 사용자를 대신해 수행하는 주체에게는 둘 중 하나의 부트랜잭션에 대한 접근 권한만 부여된다(동적 의무분리) 그런데, 이 때 정보 유출(information disclosure) 문제가 발생할 수가 있다 아래 [그림 2]와 같은 상황을 생각해 보자



[그림 2] 정보 유출문제

주체 subject1 은 Ti.1, Tj.1 객체들에 판독/기록을 허용하고, 주체 subject2 에게는 Ti 2. Tj 1 객체들에 판독/기록을 허용하는 것은 동적 의무분리 조건을 만족시킨다. 하지만, 사용자 1 의 권한(privilege)을 갖는 감염된 트로이안 목마(Trojan Horse)가 Ti.1 객체에 관한 정보를 Tj 1 객체로 전송(기록)할 수가 있다. 이 때, subject2 는 Ti 1, Ti 2 객체들 모두에 대한 정보에 대해서 판독 접근(access)이 가능하므로 정보의 유출이 발생한다 따라서 subject1 이 Tj 1 객체에 기록 연산을 위한 접근을 하지 못하게 해야 한다 위 사실에 기초하여 기록 규칙(write rule)을 다음과 같이 정의한다. 첫째, 두개 이상의 서로 다른 COI 트랜잭션 데이터 집합으로부터 객체들을 읽어왔던 주체는 절코 기록을 할 수 없다 둘째, 정확히 하나의 COI 트랜잭션 데이터 집합으로부터 객체들을 읽어온 주체는 바로 그 데이터 집합에만 닫지 기록을 할 수 있다 기록 규칙에 의해서 트로이안 목마에 의한 정보의 유출을 막을 수 있다 특히, 모든 주체가 하나의 COI 트랜잭션 데이터 집합에 판독/기록을 하도록 제한하는 것은 수용할 수 있는 제약사항이고 의무분리 요구사항에도 적합하다 물론 사용자는 공용 객체(public object)를 판독/기록하는 것 또한 허용된다

4.2 동적 의무분리를 위한 격자 구조의 설계와 해석

일반적으로 정보 흐름 정책(information flow policy)은 객체들이 격자 구조(lattice structure)상에 보안 등급(security label)이 부여될 것을 필요로 하는데, 이 것은 합리적으로 수용된다[7] 본 논문에서는 이를 바탕으로 의무분리 정책에서 정보의 흐름을 한 방향으로 제어하기 위한 격자 구조를 설계한다

Lemma 1 n개의 상호 충돌 COI(conflict of interest)(중첩)-트랜잭션 클래스가 존재한다 $COI_1, COI_2, ., COI_n$

Lemma 2 각 COI_i는 m_i 개의 상호 배타적인 부트랜잭션들을 포함한다

$COI_i = \{1, 2, .., m_i\}$, for i=1, 2, . n

Lemma 3 시스템에서 모든 객체는 보안 등급이 부여된다

$LABEL=\{[i_1,i_2, . ,i_n] \mid i_1 \in COI'_1, i_2 \in COI'_2, .. , i_n \in COI'_n\}$, where $COI'_i = COI_i \cup \{\varnothing\}$

이 보안 등급의 의미는 COI_1 트랜잭션의 부트랜잭션 $t1$, COI_2 트랜잭션의 부트랜잭션 $t2$ 등과 관련된 정보를 갖고 있는 객체를 말한다 $\varnothing$의 의미는 대응하는 COI 에서 어떤 부트랜잭션과도 연관되어 있지 않음을 뜻한다. 따라서 모든 원소가 $\varnothing$인 보안 등급은 공용 정보에 대응한다. 격자 구조를 완성하기 위해 시스템 최상위 등급을 나타내는 특별한 보안 등급 'SHIGH' 을 포함시키고, 보안 등급들 사이에 우위 관계(dominance relation) '≥' 를 정의 한다

Lemma 4. ELABEL=LABLE$\cup$ {SHIGH}

Lemma 5 $(\forall l_1, l_2 \in LABEL)$

$l_1 \geq l_2 \Leftrightarrow (\forall i_k=1, ,n) \ [\, l_1[i_k]= l_2[i_k] \ \vee \ l_2[i_k]= \varnothing \,]$

Lemma 6. $(\forall l \in ELABEL) \ [SHIGH \geq l]$

두 보안 등급의 호환(compatible), 비호환(incompatible) 관계을 정의하고 최소 상한 LUB(least upper bound) 연산자를 정의한다.

Lemma 7. $l_1, l_2 \in LABEL$ are ***compatible*** iff for all k=1, ,n, $l_1[i_k]=l_2[i_k] \vee l_2[i_k]= \varnothing \vee l_1[i_k]= \varnothing$

Lemma 8 If l_1 is ***incompatible*** with l_2 then

$LUB(l_1,l_2)= SHIGH$

Lemma 9 If l_1 is ***compatible*** with l_2 then $LUB(l_1,l_2)=l_3$ where

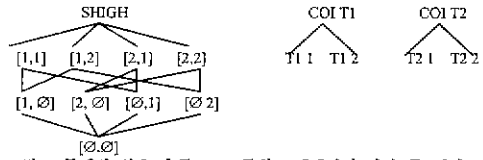
$l_3[i_k]= l_1[i_k]$, if $l_1[i_k] \neq \varnothing$

$l_3[i_k]= l_2[i_k]$, otherwise

Lemma 10 $(\forall l \in ELABEL)$

$LUB(SHIGH,l) = SHIGH$

설계된 격자에서 정보의 흐름은 아래에서 위로 향하며, 반사(reflexive), 이행(transitive), 대칭적(symmetric)인 성질을 만족한다. 모든 권한과 기록 연산은 Bell-Lapadula 모델의 simple-security 와 *-property 규칙에 따라 주체에 의해 수행된다.



[그림 3] 두개의 상호 충돌 COI 중첩_트랜잭션이 각각 두 개의 상호 배반적인 관계에 있는 부트랜잭션을 가지는 격자 구조의 예

[그림 3]의 예를 살펴보자 (COI T1={T1 1,T1.2} COI T2={T2 1,T2 2}) 만일 홍길동이 사용자로서 보안 등급 [1,1]을 가진다면(다시 말해, 홍길동에게 COI T1 에서 부트랜잭션 T1 1, COI T2 에서 부트랜잭션 T2 1 에 해당하는 역할이 배정되었을 때), 홍길동 [1,1], 홍길동 [1,Ø], 홍길동 [Ø,1], 홍길동 [Ø,Ø]과 같은 세션들이 홍길동과 연관되게 된다. 각각은 홍길동이 주어진 세션에서 로그인(login)하기를 원하는 보안 등급에 대응한다 이러한 선택 가능성이 동기인 요구사항 만족을 가능하게 한다 홍길동이 세션 [1,Ø]으로 로그인 한다고 가정하면 그 세션 동안에 생성된 모든 주체들은 보안 등급 [1,Ø]을 상속받는다 주체는 simple-security 와 *-property 규칙에 의해서 보안 등급이 [Ø,Ø]인 공용 객체를 판독, 보안 등급이 [1,Ø]인 객체를 판독/기록, 그리고 보안 등급 [1,1], [1,2] 그리고 SHIGH 인 객체를 기록하는 것이 허용된다 이 주체의 보안 등급은 [2,*] (*={Ø,1,2})인 보안 등급을 가진 어떤 객체와도 호환 관계가 성립하지 않기 때문에(Lemma 8), T1.1 과 상호 배타적인 관계에 있는 T1 2 와 연관된 객체에 대해서는 권한과 기록의 어떠한 권한을 가질 수 없다 위와 같이 위 모델을 통하여 동적 의무분리 요구사항을 만족시키면서, 또한 격자 구조를 통해 정보의 흐름을 한 방향으로 제어하므로써 정보유출 문제를 해결할 수 있다

5. 제안된 모델과 다른 모델과의 비교분석

Clark-Wilson 모델에서는 9 개의 규칙에 기반하여 무결성 보장을 위한 프레임워크를 제시했다[6] Clark 과 Wilson 은 모델의 기본적인 구성 요소들로 제약된 데이터 아이템 CDI(constrained data item)와 변환 절차 TP(transformation procedure)를 명시한다 아래와 같은 대응성을 가진다

제안된 모델	Clark-Wilson 모델
가억정보 객체	제약된 데이터 아이템(CDI)
트랜잭션	변환 절차(TP)

본 논문에서 제시한 모델에서는 시스템에서 기업정보 객체(information object)는 단지 권한이 부여된 트랜잭션들에 의해서만 수정될 수 있다는 규칙을 시행한다고 가정하고, 또한 트랜잭션의 실행은 유효 상태(valid state)를 또 다른 유효 상태로 변환시킨다고 가정한다 Clark-Wilson 모델은 비슷한 요구 사항을 또한 가지고 있다 그러나 제안된 모델에서 권한은 Clark-Wilson 모델과 중요한 점에서 서로 다르다 Clark 과 Wilson 은 의무분리 특성을 만족시키기 위해 아래의 같은 규칙을 형식화 했다 E2: "시스템은 (사용자 ID, TP_i, (CDIa, CDIb, CDIc,)) 유형의 릴레이션의 리스트를 유지해야 한다 즉 한 사용자, 한 TP, 그리고 TP 가 그 사용자를 대신해 참조하는 데이터 객체들의 관계를 나타낸다 이 규칙에 따라 단지 릴레이션들에 명시된 실행들만이 수행된다는 것을 보장해야 한다."

Clark-Wilson 모델은 동적 의무분리를 위해 이러한 릴레이션들이 명시적으로 유지되어야 함을 암시하고 있고 더욱이 동적으로 갱신되는 릴레이션들의 집합을 유지해야 한다 역할 기반의 접근제어 시스템에서는 개개의 사용자들에 대해 권한을 명세하는 릴레이션들의 리스트를 유지하기 보다는 역할의 개념으로 대체하므로써 규칙 E2 를 단순화시킬 수 있다 이 논문에서 제시한 세션에 기반한 동적 의무분리 기법에서는 모든 데이터 객체는 생성될 때 제시한 공리에 따라 보안 등급이 부여되고 격자 구조가 만들어 진다 실행시에 내부적으로 모든 사용자는 권한이 부여된 트랜잭션 리스트를 가지게 되는데 이것은 실제로 그 사용자를 대신해 수행하는 주체의 보안 등급으로 간주된다 그 비교나서 설계된 격자 위에서 Bell-Lapadula 의 규칙에 따라 객체에 대한 판독과 기록 연산을 하게 된다 따라서 사용자는 단순히 단지 트랜잭션을 수행할 수 있는 역할에 지정되었는지를 확인하는 절차를 거치면 된다

6. 결론 및 추후 연구과제

본 논문에서는 기업 환경에 적합한 역할 기반의 접근제어 시스템에서 데이터베이스 내에 저장된 데이터에 대한 권한 없는 접근, 고의적인 파괴 및 변경으로 인한 무결성 침해로부터 데이터베이스를 보호하기 위해 의무분리의 원리를 제안하였다 그리고 제안된 원리를 기반으로 하여 작용 대상으로 상호 배타적인 관계에 있는 부트랜잭션들을 포함한 중첩_트랜잭션을 생각해 보았으며, 여기에 동적 의무분리 요구사항을 만족시키기 위해서 주체_세션 기반에서 새롭게 해석하였다 제안된 이 기법은 시스템 운영의 유연성을 향상시키고 역할의 수를 줄일 수 있어 관리상의 단순화를 제공할 수 있다 여러 트랜잭션들이 동시에 실행되는 환경에서 본 논문이 제안한 세션기반의 동적 의무분리 원리를 적용하였다 이때 발생할 수 있는 정보의 유출문제를 해결하여 정보의 흐름을 한 방향으로 제어하기 위해서 격자 구조를 설계하고 이를 바탕으로 해석하였다. 앞으로, 제안한 모델을 보다 정형화된 방식으로 기술하고 제안한 모델과 기존에 제시되었던 Clark-Wilson 모델[4] 등과 같은 무결성에 보장을 위한 일반화된 모델, 그리고 의무분리 모델들[Sandhu 모델[3] 등]과의 관계를 비교 분석하고 평가하는 연구가 더 필요하다.

■ 참고문헌

[1] David Ferraiolo and Richard Kuhn "Role-Based Access Control", Proceedings of 15th National Computer Security Conference, 1992

[2] 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 Conference, 1995 pages 241~248

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

[4] 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, 1987, pages 184~194

[5] Ravi Sandhu and Venkata Bhamidipati, "The ARBAC97 Model for Role-Based Administration of Roles Preliminary Description and Outline", preceedings of Second ACM Workshop on Role-Based Access Control, Nov, 1997

[6] 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

[7] Ravi Sandhu, "A lattice interpretation of the chinese wall policy", Proceedings of the 15th NIST-NCSC National Computer Security Conference, pages 221~235