역할 격자구조를 이용한 망관리 시스템들의 보안 상호운용

서 재 현[†] · 김 태 연[††] · 노 봉 남[†††]

요 약

망이 대규모화되고 복잡해짐에 따라 하나의 망관리자에 의해 전체 망관리 시스템을 관리하는 어려움은 여러 망관리자에 관리 행위가 분산되어 수행되어야 한다. 또한 서로 다른 망관리 정책을 사용하는 각각의 망관리 시스템 사이에는 보안을 유지하며 상호운용성을 제공해야 한다. 본 논문에서는 각 망관리 시스템에 역할 격자구조를 보유하게 하고 새로운 역할의 요구시 동적으로 역할을 부여하는 알고리즘을 통하여 망관리 시스템간의 보안 상호운용성을 제공하도록 하였다. 또한 역할 격자구조에 하나의 역할과 다른 역할 사이의 상호 관련성을 고려하여 시스템의 상태에 따라 역할들의 관리 권한을 동적으로 변경하여 비밀성과 무결성이 유지되도록 하는 여러가지 보안규칙을 제안하였다. 마지막으로 각각의 보안 규칙들을 ECA 규칙을 이용하여 표현하였다.

Security Interoperation of Network Management Systems using Role Lattices

Jae Hyeon Seo[†] · Tae Yeon Kim[††] · Bong Nam Noh[†††]

ABSTRACT

As the size and complexity of networks increase, it is difficult to manage the whole network using single network manager, thus it is more reasonable to manage the network using several network managers distributed on the network. Security interoperability should be supported among network management systems(NMSs) that use different management policies. In this paper, an algorithm that makes it possible to take a role into a role lattices is suggested to provide security interoperability among NMSs that have their own role lattices. In addition, security constraints are proposed to maintain the confidentiality and integrity of information by dynamically modifying the access rights of roles as the state of a system changes. Also, the security constraints are expressed using ECA rules in this paper.

1. 서 론

컴퓨터를 사용하는 사용자의 수가 급증하고 상호 독립적으로 존재하던 컴퓨터들이 통신망을 통하여 상호 연동 됨에 따라 전체적인 통신 망의 규모가 점점 커지고 복잡해졌다. 또한 망을 이용하는 사용자들의 요구 사항이 다양해져 이를 효율적으로 관리해 줄 수 있는 망관리 시스템이 통신 망의 운용에 필수적인 요소이다[5, 6].

망을 효율적으로 관리하기 위해서는 망관리 자원의 개념적인 정보 저장소인 관리 정보베이스에 대한 안전한 접근 제어가 행해져야 한다. 자율적 접근 제

† 정 회 원:목포대학교 컴퓨터공학과 전임강사
†† 정 회 원:서남대학교 전산정보학과 전임강사
††† 종신회원:전남대학교 전산학과 교수
논문접수:1996년 1월 13일, 심사완료:1996년 8월 30일

어는 사용자 신원을 기반으로 객체에 대한 접근을 통제하는 기법이며, 강제적 접근 제어는 사용자의 인가 등급과 객체의 보안등급을 기반으로 접근을 통제하는 기법이다.

역할기반 접근 제어에서는 여러 명의 사용자는 역할이라고 하는 클래스들로 그룹화 되어 지며 역할은 조직체에서 사용자의 활동 위치에 따라 결정된다. 역할 권한부여는 역할에 사용자 할당과 역할의 객체에 대한 접근 권한 부여의 두 단계로 나누어 수행함으로써 사용자의 접근 권한 변경 등으로 인한 접근 권한 부여 관리에 효율적이다. 역할기반 접근 제어는 자율적 접근 제어보다는 안전한 정보의 흐름을 보장하고, 강제적 접근 제어보다는 융통성 있는 접근 제어를 제공한다.

망관리에서 통신망의 계획, 설계와 망자원의 추가, 변경 등으로 인한 망 구성의 확장은 중요한 문제이다. 소규모 망에서는 이러한 문제를 하나의 망관리자가 해결할 수 있지만 망이 복잡해지고 다양화됨에 따라 하나의 망관리자가 망관리 시스템을 관리할 경우 실시간 처리가 어렵고 망의 운영에 효율성을 보장할 수 없는 등의 많은 문제점을 야기한다[15]. 그러므로 대규모 망관리 시스템에서 망관리 기능을 수행하는 관리자를 구분하여 관리자들을 계층화하고 상위 계층의 관리자는 관리 기능을 하위 계층의 관리자에 위임하여 영역별 관리자에 의해서 망관리를 수행하는 연구가 필요하다. 또한 망관리 시스템들 사이에 관리 정보를 공유하기 위해서 상호운용성을 제공할 수 있는 연구가 수행되어야 한다.

본 논문에서는 각 망관리 시스템은 사용자들의 역할에 따라 하위역할의 접근권한이 상위역할에 상속되는 역할 격자구조를 유지하여 망관리 시스템들에서 새로운 역할의 요구시 동적으로 역할을 생성, 삽입하여 망관리 시스템간의 보안 상호운용성을 제공하도록 하였다.

분산망 환경에서는 사용자와 자원이 널리 분산되어 있어 사용자와 자원 사이에 복잡한 접근 제어 메카니즘이 요구된다. 복잡한 분산망 환경에서는 사용자와 관리객체에 대한 접근 권한의 관리 뿐만 아니라 사용자들 사이의 상호 관련성을 고려하여 접근권한을 관리함으로써 관리객체의 비밀성과 무결성을 보장할 수 있는 메카니즘이 필요하다.

본 논문에서는 역할 격자구조에 이행성, 상호배제와 임무분리 등의 보안규칙에 따라 하나의 역할과 다른 역할 사이의 상호 관련성을 고려하여 시스템의 상태에 따라 역할들의 관리권한을 동적으로 변경하여 비밀성과 무결성이 유지되도록 하였다. 마지막으로 각각의 보안 규칙들을 ECA(Event-Condition-Action) 규칙을 이용하여 표현하였다.

2. 관련 연구

각 조직체들에서는 서로 다른 관리정책을 사용하여 망자원과 서비스를 관리하기 위해 망관리 시스템을 각각 구축하고 있다. 효율적인 정보의 공유를 위해 어떤 조직의 망관리 시스템에서 다른 조직의 망관리 시스템과 상호운용이 요구되지만 이는 매우 복잡한 문제이다. 또한 하위역할에서 상위역할로 상속되는 정상적인 접근권한 관리뿐만 아니라 분산 환경에서는 다른 주체와의 상호 관련성을 고려하여 접근권한을 관리함으로써 관리객체의 비밀성과 무결성을 보장할 수 있다.

망관리 정보베이스를 안전하게 관리하기 위한 접근제어 정책중에 하나인 역할기반 접근제어 정책은 자율적 접근제어 보다는 강력하고 강제적 접근제어 보다는 융통성을 제공하는 접근제어 방법으로 최근에 많은 연구가 진행중에 있다. 역할 기반 접근 제어 정책을 사용하는 데이타베이스들 사이의 상호운용성을 보장하기 위한 연구가 진행되었다[14]. 이 연구에서는 객체지향 데이타베이스의 클래스 상속구조에 새로운 클래스를 통합하는 연구를 역할 격자구조에 적용하여 새로운 역할을 동적으로 기존의 격자구조에 삽입하여 데이타베이스 사이에 상호운용성을 제공할 수 있는 방법을 제공하였다. 이 논문은 역할 격자구조를 이용하여 데이타베이스들 사이에 상호운용성을 제공하기 위한 연구이다. 그러나 분산망 환경의 많은 상업적 응용에서 요구되는 역할들 사이의 관련성에 관한 연구는 하지 않았다. 분산 환경에서는 상호운용성 뿐만 아니라 동적으로 변하는 역할들 사이의 상호 관련성을 고려하므로 써 비밀성과 무결성을 보장할 수 있는 메카니즘이 제공되어야만 한다.

망관리 정보베이스가 안전하게 관리되기 위해서는 망관리자의 접근을 효율적으로 제어할 수 있어야 할

뿐만아니라 망에서 발생하는 사건(event)의 발생도 적절하게 통제할 수 있어야 하므로 망관리 분야에 능동 데이타베이스를 적용하는 연구가 진행중에 있다. [8]에서는 망관리 데이타베이스에서 능동개념과 시간개념을 표현하기 위한 모델을 제안하고 있다. 이 논문에서는 사건들을 기술할 수 있는 사건 기술 언어를 제시하였고, 페트리네트를 이용하여 복합 사건들을 모형화하였다. 사건 기술 언어들은 복합 사건이 발생하였을 때, 사건들 사이의 시간적인 관계를 표현하는데 효과적으로 사용되어질 수 있다. 또한 ECA 규칙을 이용하여 이들의 능동적인 특성을 표현하였다. 이 논문은 망에서 발생하는 복합사건에 대해서 능동개념과 시간개념을 적용하여 표현하였을 뿐이며 망관리 정보베이스에 대한 효율적인 접근 제어에 관한 연구는 진행되지 않았다.

망관리 시스템들 사이에 안전한 망관리를 위해서는 망관리 시스템 사용자의 망 접근 뿐만아니라 망 사용자들 사이의 관련성을 고려하여 망관리 정보베이스에 대한 접근제어를 수행 함으로써 망관리 정보베이스의 비밀성과 무결성을 유지할 수 있다. [11]은 분산 환경에서 복잡한 접근 제어 정책들을 모형화하였다. 이 논문에서는 정상적인 접근 제어 정책에서 벗어나는 예외적인 사항들을 정의하므로써 정보의 무결성과 비밀성을 유지할 수 있는 방안을 제시하였다. 또한, 부울 표현에 기반을 둔 접근 제어(BEAC: Boolean Expression based Access Control) 모델을 제안하고, 이 모델을 이용하여 예외 사항들을 정의하고 있다. 그러나 이 논문에서는 시간에 따라 동적으로 변하는 시스템의 상황을 표현할 수 있는 적절한 방안을 제시하지 못하고 있다.

자율적 접근 제어와 강제적 접근 제어는 사용자의 역할 변경 문제를 해결하지 못하고 있는데, [7]에서는 역할기반 접근 제어에서 사용자의 특권이 변경되었을 때 능동적으로 보안이 유지되기 위해 적용될 수 있는 메카니즘을 제안하고, 능동 데이타베이스에 적용될 수 있는 방법을 제안하고 있다.

본 논문에서는 역할기반 접근 제어 정책에 따라 각각 고유의 역할 격자구조를 가지고 있는 망관리 시스템들 사이의 상호운용성을 제공하기 위하여, 새로운 역할을 다른 역할 격자구조에 삽입할 수 있는 알고리즘을 제시하였다. 그리고 이를 확장하여 두개의 역할 격자구조를 하나의 역할 격자구조로 통합할 수 있는 알고리즘도 고안하였다. 뿐만 아니라, 역할들이 객체를 접근할 때 정보의 비밀성과 무결성 등을 보장할 수 있는 여러가지 보안 규칙들을 제안하였고, 이들 각각을 ECA 규칙들을 이용하여 정의하였다.

3. 역할기반 보안 정책

보안 정책의 개념은 매우 광범위하며, 많은 분야에서 각각의 의미를 갖고서 다른 방법들로 이용되고 있다. 가장 많이 이용되는 방법은 특정한 정보 시스템이 조직의 보안 요구사항을 지원하기 위하여 처리되는 과정을 서술하는 시스템 보안 정책이며, 이는 접근 제어 정책으로 정의된다. 접근 제어 정책의 목적은 컴퓨터, 통신 그리고 정보 자원에 대한 권한이 부여되지 않은 접근을 방지하는 것이다.

접근 제어 정책에는 자율적 접근 제어, 강제적 접근 제어 및 역할기반 접근 제어 등이다. 자율 접근 제어는 사용자 신원을 기반으로 객체에 대한 접근을 통제하는 기법이며, 강제적 접근 제어는 사용자의 인가등급과 객체의 보안등급을 기반으로 접근을 통제하는 기법이다.

역할기반 접근제어에서는 데이타 또는 객체를 몇개의 범주로 나누었으며, 여러 명의 사용자는 역할이라고 하는 클래스들로 그룹화되어 진다. 역할은 어떤 조직체의 사용자 임무를 여러 개의 영역으로 분할해 놓은 것으로서 역할 이름과 범주에 접근할 수 있는 권한으로 구성되어 있다.

【정의 1】 역할(role)

역할은 역할 이름, 그 역할이 접근할 수 있는 객체 및 그 객체에 대한 접근 모드의 쌍으로 이루어진 리스트로 구성된다.

R = (R_id, P_list) 여기서 R_id는 역할 이름이고 P_list는 특권 리스트이다.

역할은 관리를 단순화하고 그에 따라 접근 권한을 단순화해 주는 적당한 형태로 조직될 수 있다. 역할들은 함축적인 역할 상호간의 연관성에 따라 링크로 연결되며, 역할의 격자구조를 형성한다. 역할 격자구조를 다음과 같이 정의한다.

【정의 2】 역할 격자구조(role lattice)

조직체의 각 역할을 상호 연관성에 따라 권한이 높은 상위역할과 하위역할이 링크로 연결된 구조를 역할 격자구조로 정의한다.

역할 격자구조에서 상위역할은 자신의 접근 권한을 하위 관리자 역할에 위임하여 하위 역할은 자신의 영역에 대한 접근 권한을 관리하며, 점진적으로 다시 관리권한을 하위역할에 위임할 수 있다. 역으로 하위역할의 관리정보베이스에 대한 접근 권한 속성은 상위역할에 묵시적으로 상속되어진다. 이렇게 역할 격자구조에 객체지향의 상속개념을 적용하여 망관리 정보베이스의 접근권한의 관리를 효율적으로 수행할 수 있다.

역할 격자구조에서 각 역할 클래스 사이의 관계는 각 역할 클래스가 보유하는 접근권한 속성의 포함관계에 의해 표현할 수 있다. 역할 격자구조에서 부분집합 클래스와 상위집합 클래스를 다음과 같이 정의하였다.

【정의 3】 역할 클래스 관계

두개의 역할 클래스 RC_1, $RC_2 \in RC$이고 A가 속성의 집합일때, $(\forall a \in A)((a \in RC_1) \rightarrow (a \in RC_2))$이면 RC_1을 RC_2의 부분집합 클래스, RC_2를 RC_1의 상위집합 클래스로 정의한다.

임의의 역할과 상위역할 클래스와의 관련성을 위해 최소상한 역할을 다음과 같이 정의한다.

【정의 4】 최소상한 역할

역할의 접근권한 속성들을 포함하는 속성을 갖는 상위집합 클래스들 중에서 가장 적은 접근권한이 부여된 역할을 최소상한 역할로 정의한다.

또한 임의의 역할과 하위 역할 클래스와의 관련성을 위해 최소상한 역할을 다음과 같이 정의한다.

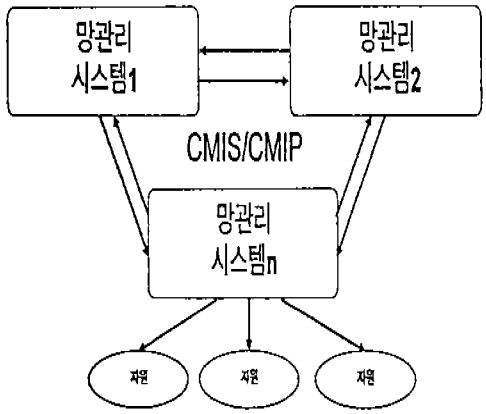
【정의 5】 최대하한 역할

역할의 접근권한 속성에 포함되는 속성을 갖는 부분집합 클래스들 중에서 가장 많은 접근권한이 부여된 역할을 최대하한 역할로 정의한다.

4. 분산 망관리 시스템간의 보안 상호운용

4.1 분산 망관리 시스템

망이 대규모화되고 복잡해짐에 따라 하나의 망관리자에 의해 망관리 시스템을 관리하기가 힘들므로 여러개의 망관리자에 관리행위가 분산되어 수행되어야 한다. 또한 서로 다른 망관리 정책을 사용하는 망관리 시스템 사이에는 각 망관리 시스템의 보안 위반 없이 관리정보의 상호운용성을 제공해야 한다.



(그림 1) 망관리 시스템 상호운용 모델
(Fig. 1) Interoperation Model of NMSs

분산망 환경에서 하나의 망관리 시스템은 그 자신이 관리하는 망(공중망, PABX, LAN, MAN)에서 뿐만 아니라 또다른 망관리 시스템의 관리자원을 공유할 수 있도록 있도록 상호운용성을 제공하여야 한다. 망관리 시스템 사이에 상호운용성을 제공하기 위해서는 두 망관리 시스템 사이에 지원되는 프로토콜, 관리기능과 관리객체에대한 공통의 이해가 선행되어야 한다. 이러한 상호운영을 위한 정보는 X.500 디렉토리에 중앙집중식으로 저장되거나 각 망관리 시스템에 관리객체의 인스턴스로 저장되어질 수 있다.

본 논문에서는 객체 지향 개념을 기반으로 관리객체를 정의하는 OSI망을 기반으로 하고, 망관리 시스템 사이에 OSI의 망관리 프로토콜인 CMIS/CMIP을 사용하고 있다. CMIS가 제공하는 서비스는 M-GET, M-SET, M-ACTION, M-CREATE, M-DELETE, M-CANCEL-GET, M-EVENT-REPORT등의 일곱개의 서비스 요소로 구성되며 CMIP은 11개의 프로토콜 데이타 단위로 구성되어 망관리 정보를 전송하기 위

한 프로시져와 CMIS를 위한 구문을 정의하고 있다. 또한 본 논문에서는 망관리 시스템 사이에 안전한 상호운용을 위해서 각 망관리 시스템에서는 망관리를 위한 관리자 역할들의 격자구조를 인스턴스로 유지하는 역할 기반 접근 제어 정책을 사용하고 있다.

분산화된 대규모 망관리 시스템의 관리를 기존의 국부적인 소규모 망관리에서 처럼 하나의 관리자가 수행하는 것은 효율적인 망관리 측면에서 바람직하지 못하다. 따라서 망에 발생한 문제들을 신속하게 처리하기 위해서는 망관리 시스템을 영역별로 분할하여 관리하고, 이렇게 분산 관리되는 망을 사용자에게는 단일 망을 관리하는 것과 같은 투명성을 제공할 수 있는 망관리 구조가 필요하다.

본 논문에서는 망관리 시스템을 지리적 여건 또는 관리 기능에 따라서 관리 영역으로 분할하여 이러한 관리 영역을 관리하는 관리자 역할을 할당하고 점진적으로 관리 영역을 다시 하위 영역으로 구분하여 각각의 관리자 역할을 할당할 수 있다. 또한 관리자 역할들 사이에 링크로 연결되는 역할 계층구조를 형성하여 관리상의 효율성을 보장할 수 있다.

망관리 시스템 사이에 역할을 통한 상호운용을 위해서 각 망관리 시스템 사이에 역할 클래스의 특성을 기술하는 속성들에 대한 이해가 선행되어야 한다. 각 망관리 시스템에서는 자신의 망관리 환경에 적합한 역할 격자구조를 유지하며 다른 망관리 시스템의 사용자 역할로부터 관리객체에 대한 연산의 요구시 역할 격자구조의 보안성을 유지하며 새로운 역할은 기존의 역할 격자구조에 삽입되어야 한다.

또한 분산망 환경의 많은 상업적인 응용에서의 보안 요구 사항은 역할 격자구조의 특징을 파괴할 수 있다. 즉 사용자의 관리객체에 대한 접근권한 부여는 정상적인 접근권한 규칙뿐만 아니라 다른 사용자와의 상호 관련성을 고려하여야 한다. 관리객체의 무결성을 유지하기 위하여 하나의 역할은 다른 역할과의 관련성에 따라 접근 권한이 동적으로 변경될 수 있다.

다음 장에서는 역할 격자구조에 역할 삽입을 통한 망관리 시스템 사이에 보안 상호운용성을 제공하지 위한 연구 및 망관리 시스템의 보안 유지를 위한 연구를 하였다.

4.2 역할 삽입을 통한 망관리 시스템의 상호운용

역할기반 접근 제어 정책은 사용하는 각 망관리 시스템들은 각각의 역할 격자구조를 가지고 있으며 이러한 각각의 망관리 시스템 들간의 상호운용을 위해서는 서로 다른 역할 격자구조들 사이의 관련성이 알려져 있어야 한다. 하나의 방법은 두 망관리 시스템들의 관리자들이 미리 하나의 격자구조내에 공유하는 서로 다른 격자구조를 포함시키는 방법이다.

그러나 많은 응용에 있어서 필요로 하는 공유 정보를 미리 예측하기는 매우 어려운 문제이다. 즉 하나의 망관리 시스템에서 서로 다른 망관리 시스템에 접근을 어떤 사용자가 행할 것인가를 미리 예측할 수는 없다. 그러므로 서로 다른 격자구조들을 실행시간 병합이 필요하다. 즉 역할 격자구조들의 동적인 병합을 통하여 보안 위반사항의 발생 없이 정보의 공유가 이루어져야 한다.

새로운 사용자나 프로그램이 망관리 정보베이스에 접근을 원할 때 새로운 주체에 대한 역할을 결정하는 인증 메카니즘이 필요하다. 예를 들어 새로운 주체에게 인증서를 제출하게 하고 인증 서버로 하여금 인증서의 내용에 따라 새로운 주체가 어떤 역할에 해당하는지 인증하게 할 수 있다. 그리고 사용자 역할기반 접근 제어 시스템에서는 그 주체의 역할에 따라 망관리 정보베이스에 대한 접근 제어가 행하여지며 요청된 역할이 역할 격자구조에 존재하지 않으면 새로운 역할들을 동적으로 생성하여 기존의 역할 격자구조에 병합하는 작업이 필요하다.

객체지향 클래스 계층구조에서는 격자구조내에서 클래스와 관련 있는 속성들을 기반으로 클래스들 사이의 관련성을 표현한다. 객체지향 클래스 계층구조에서 병합에 사용되는 방법을 역할 격자구조 병합에도 적용할 수 있다.

역할 클래스 격자구조에 역할 클래스를 병합하는 데는 클래스 사이의 상속성과 하위 클래스 사이의 관련성에 따라 가장 적절한 위치를 찾는 것이 가장 중요한 문제이다. 이를 위해서 본 논문에서는 역할 클래스들의 권한부여 속성 사이의 포함관계를 결정하여 역할 격자구조 상의 적절한 위치를 지정한다. 즉 역할 격자구조에 삽입되는 역할은 최소상한 역할의 하위에 최대하한 역할의 상위에 삽입되어야 한다. 그러므로 새로운 역할 클래스는 기존의 역할 격자구조의 보안 특성을 유지시키며 삽입될 수 있다.

가. 【격자구조에 역할 삽입 알고리즘】

Input: L, r;

Output: L'

L: 역할 격자구조

L': 새로운 역할 격자구조

r: 격자구조에 삽입되는 역할

u, v, t, b: 역할 격자구조내의 병합

【1】COPY($L \rightarrow L'$)

/* 격자구조 L을 새로운 격자구조 L'에 복사 */

【2】EXECUTE dfs FROM root(L')

/* 격자구조 L'의 뿌리 노드로부터 깊이 우선 검색을 수행 */

【3】SEEK last node u IN L' such that $Attrib(u) \supseteq Attrib(r)$

MAKE link FROM r to u

/* 새로 삽입하는 역할 r과 L'의 역할 노드를 비교하여 역할 r을 포함하는 마지막 노드 u를 찾아서 u를 역할 r의 상위역할로 삽입 */

【4】SEEK first node v IN L' such that $Attrib(v) \subseteq Attrib(r)$

MAKE link from v TO r

/* 새로 삽입하는 역할 r의 부분집합인 첫번째 역할 노드 v를 찾아서 v를 역할 r의 하위역할로 삽입 */

【5】REPEAT 【2】~【4】 UNTIL find proper position

/* 적절한 삽입 위치를 찾을때까지 【2】【3】【4】를 반복 */

【6】IF NOT FOUND, MAKE link ((FROM r TO $t(L')$) $\wedge$ (FROM $b(L')$ TO r)

/* 만약 적당한 삽입할 적당한 위치를 찾지 못하면, top 노드 t를 역할 r의 상위역할로, bottom 노드 b를 역할 r의 하위역할로 연결 */

위 알고리즘에서는 기존의 역할 격자구조에서 접근권한이 많은 뿌리 노드로부터 하향적으로 순회하면서, 상위역할과 삽입하는 역할 클래스의 권한부여 속성을 비교한다. 이렇게 하여 가장 마지막 상위역할을 찾고 마지막 상위역할 노드로부터 삽입할 역할 노드로 연결한다. 또한 새로 삽입하는 역할의 부분집합인 첫번째 역할 노드를 찾아 삽입하는 역할로부터 부분집합인 첫번째 역할 노드로 연결한다. 그러므로 기존 격자구조의 권한 부여 속성을 유지하며 새로운 역할 클래스를 삽입할 수 있다.

역할기반 접근 제어 정책을 사용하는 각 망관리 시스템들은 각각의 역할 격자구조를 가지고 있으며 이러한 각각의 망관리 시스템들 간의 상호운용을 위해서는 서로 다른 역할 격자구조들 사이의 관련성이 알려져 있어야 한다. 망관리 시스템의 역할 격자구조와 또 다른 망관리 시스템의 역할 격자구조의 병합도 유사하게 수행할 수 있다. 역할 격자구조 L_1에 새로운 역할 격자구조 L_2를 동적으로 병합하는 알고리즘은 다음과 같다.

나. 【역할 격자구조 병합 알고리즘】

Input: L_1, L_2

Output: L_3

L_1, L_2: 병합을 위한 역할 격자구조

L_3: 새로운 역할 격자구조

$t(L_1)$, $t(L_2)$: 역할 격자구조 L_1, L_2의 TOP

$b(L_1)$, $b(L_2)$: 역할 격자구조 L_1, L_2의 BOTTOM

【1】COPY(L_1, $L_2 \rightarrow L_3$)

/* 격자구조 L_1, L_2의 격자구조를 L_3에 복사 */

【2】EXECUTE dfs FROM root(L_2)

/* 격자구조 L_2의 뿌리 노드로 부터 깊이 우선 검색을 수행 */

【3】SEEK last node u IN L_2 such that $Attrib(u) \supseteq Attrib(t(L_1))$

MAKE link FROM $t(L_1)$ to u

/* 새로 삽입하는 격자구조 L_1의 top 노드를 L_2의 노드들과 비교하여 top 노드를 포함하는 마지막 노드 u를 찾아서 u를 격자구조 L_1의 top 노드의 상위 역할로 연결 */

【4】SEEK first node v IN L_2 such that $Attrib(v) \subseteq Attrib(b(L_1))$

MAKE link from v TO $t(L_1)$

/* 새로 삽입하는 격자구조 L_1의 bottom 노드를 L_2의 노드들과 비교하여 bottom 노드에 포함되는 첫 번째 노드 u를 찾아서 u를 격자구조 L_1의 bottom 노드의 하위역할로 연결 */

【5】REPEAT 【2】~【4】 UNTIL find proper position

/* 적절한 삽입 위치를 찾을 때까지 【2】~【4】를 반복 */

【6】IF NOT FOUND, MAKE link ((FROM $t(L_1)$ TO $t(L_2)$) ∧ (FROM $b(L_2)$ TO $b(L_1)$)
/* 만약 삽입할 적당한 위치를 찾지 못하면, 격자구조 L_2의 top 노드 t를 격자구조 L_1의 top 노드의 상위역할로, 격자구조 L_2의 bottom 노드 b를 격자구조 L_1의 bottom 노드의 하위역할로 연결 */

망관리 시스템들은 보안 상호운용을 위해 역할 격자구조를 유지하고 관리자 역할에서 관리 정보베이스에 대한 접근 요구시 역할이 존재하면 역할에 주어진 접근권한에 따라 망관리 정보베이스에 접근할 수 있지만 관리자 역할이 역할 격자구조에 존재하지 않으면 수행 시간에 동적으로 역할이 역할 격자구조에 삽입되어야 한다. 역할 기반 접근제어 정책을 기반으로 하나의 망관리 시스템의 보안 특성은 유지시키며 다른 망관리 시스템의 관리자 역할을 기존의 역할 격자구조의 가장 적절한 위치에 삽입하므로 써 두 망관리 시스템 사이에 보안 상호운용성을 제공할 수 있다.

본 논문에서 제안하는 알고리즘은 역할 격자구조에 역할의 삽입시 삽입되는 역할은 최소상한 역할의 하위역할로 최대하한 역할의 상위역할로 연결되어 기존의 역할 격자구조의 보안 위반사항 없이 상호운용성을 제공할 수 있다.

4.3 분산 망관리 시스템의 보안 유지 규칙

분산망 환경에서는 사용자와 자원이 널리 분산되어 있어 접근권한 부여를 위해 복잡한 접근제어 메카니즘이 요구된다. 분산망 환경에서 역할은 접근제어의 관리를 쉽게 해주는 적당한 형태로 조직될 수 있으며 객체지향의 일반화와 세분화를 기반으로 많은 응용에서 역할 사이에 격자구조를 형성하여 상위 역할과 하위 역할 사이에 상속개념이 적용될 수 있다. 즉 하위 역할의 명시적으로 부여된 접근권한은 상위 역할로 상속되어 접근권한의 관리를 단순화시킨다.

그러나 많은 상업적인 응용에서 보안 요구 사항은 역할 격자구조의 특징을 파괴할 수 있다. 즉 사용자의 관리 객체에 대한 접근권한 부여는 정상적인 접근권한 규칙뿐만 아니라 다른 사용자와의 상호 관련성을 고려하여야 한다. 비밀성을 중요시하는 군사적인 보안 모델과는 달리 상업적인 보안 모델에서는 관리 객체의 무결성 유지가 중요한 관심 사항이다. 본 절에서는 역할 격자구조에서 무결성과 비밀성 유지를 위해 필요한 보안규칙들을 제안하였다.

역할 격자구조에서 정상적인 접근 권한 규칙을 위반하는 보안 요구 사항은 이행성 보안 규칙, 상호배제 보안 규칙과 임무분리 보안 규칙 등이다. 이들의 예가 그림 2에 제시되었다.

이행성 보안 규칙은 하위의 역할 클래스에서 특정 객체에 접근하고자 할 때, 상위 역할 클래스를 통해서만 수행이 가능하게 하는 보안규칙이다. 즉 특정 관리 객체에 대한 연산이 신뢰성 있는 상위 역할에 의해서만 수행 가능하게 하여 비밀성과 무결성을 유지하게 하는 보안 규칙이다. 상호배제 보안 규칙은 두 개 이상의 역할이 하나의 객체에 대하여 접근하려고 할 때, 이미 하나의 역할이 그 객체에 대하여 접근 연산을 수행하였다면 다른 역할들은 더 이상 그 객체에 대하여 접근을 할 수 없도록 하는 규칙이다. 또한 임무분리 규칙은 역할들이 특정 객체에 접근하려고 할 때, 이미 이전에 그 객체에 접근한 적이 있다면 더 이상 그 객체에 대하여 접근할 수 없도록 하는 보안 규칙이다.

본 논문에서는 관리 객체의 비밀성과 무결성을 유지하기 위한 보안 규칙들을 정의하고 이들을 다음과 같이 ECA 규칙으로 표현하였다. ECA 규칙 내에서 사용되는 함수 및 용어들의 의미는 다음과 같이 정의할 수 있다.

□ R_i, $1 \le i \le n$
접근 주체가 되는 관리자의 역할

□ O_i, $1 \le i \le n$
접근 대상이 되는 관리 객체

□ *READ*($R_i \rightarrow O_j$), $1 \le i, j \le n$
관리자 역할 R_i의 관리 객체 O_j에 대한 읽기(read) 연산

□ *WRITE*($R_i \rightarrow O_j$), $1 \le i, j \le n$
관리자 역할 R_i의 관리 객체 O_j에 대한 기록(write) 연산

□ *PERMIT*(*Op*)
관리 연산(*Op*)을 허용하는 함수

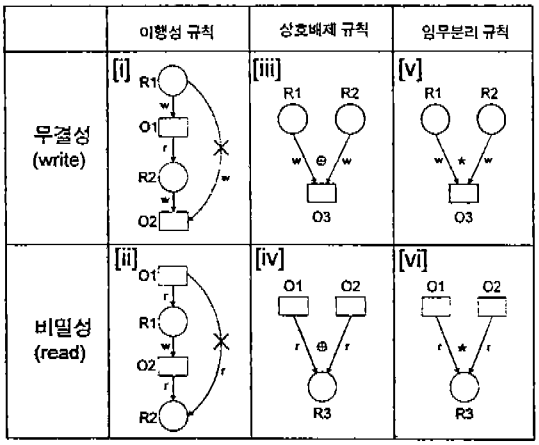
□ *PREVENT*(*Op*)
관리 연산(*Op*)을 금지하는 함수

□*DEFINED*(*Op*)

관리 연산(*Op*)이 수행 가능한지를 검사하는 함수로서, TRUE 또는 FALSE 값을 반환

□*OCCURRED*(*Op*)

관리 연산(*Op*)이 발생하였는지를 검사하는 함수로서, TRUE 또는 FALSE 값을 반환



(그림 2) 보안 규칙의 분류
(Fig. 2) Classification of Security Constraints

이행성 보안 규칙은 무결성과 비밀성에 따라 보안 규칙 1과 보안 규칙 2처럼 정의될 수 있다.

가. 【보안 규칙 1】 이행 무결성 보안 규칙

임의의 역할의 특정 객체에 기록연산은 반드시 하위 역할을 통해서만 가능하다.

일반적인 경우에는 역할이 가지고 있는 권한에 따라 특별한 제약없이 임의의 역할이 특정 객체에 기록연산을 수행할 수 있다. 그러나 이행 무결성 보안 규칙은 그런 경우에 일어날 수 있는 보안 위반 사항의 발생을 방지하여 주므로써, 객체의 무결성을 보장해 주기 위하여 필요하다. 그림 4.8 [i]의 이행 무결성 보안 규칙을 ECA 규칙을 이용하여 다음과 같이 정의할 수 있다.

[i]이행 무결성 보안 규칙 ($\sim E_w$)

E:*WRITE*($R_1 \rightarrow O_2$)

C:$\sim$*DEFINED*(*WRITE*($R_1 \rightarrow O_2$)) $\wedge$ *DEFINED* (*WRITE*($R_1 \rightarrow O_1$) $\wedge$ *READ*($R_2 \rightarrow O_2$) $\wedge$ *WRITE*($R_2 \rightarrow O_2$))

A:*PERMIT*((*WRITE*($R_1 \rightarrow O_1$), *READ*($R_2 \rightarrow O_2$), *WRITE*($R_2 \rightarrow O_2$))

위에 정의한 이행 무결성 보안 규칙은 역할 R_1이 객체 O_2에 기록연산을 수행하고자 하는 사건(*WRITE*($R_1 \rightarrow O_2$))이 발생한 경우의 예이다. 일반적인 경우에는 역할 R_1이 객체 O_2에 기록 접근권한을 가지고 있기 때문에 기록연산이 가능하지만, 이 경우에는 무결성 유지를 위한 보안 규칙의 적용을 받아 역할 R_1이 객체 O_2에 직접 기록연산을 수행할 수 없고, 반드시 역할 R_2를 통해서만 가능하다. 즉, 먼저 역할 R_1이 객체 O_1에 기록연산을 수행(*WRITE*($R_1 \rightarrow O_1$))하고, 역할 R_2가 객체 O_1으로부터 그 정보를 읽은 다음 (*READ*($R_2 \rightarrow O_2$)), 역할 R_2가 다시 객체 O_2에 그 정보를 기록(*WRITE*($R_2 \rightarrow O_2$))하므로써 정보의 무결성을 보장하여 줄 수 있다.

나. 【보안 규칙 2】 이행 비밀성 보안규칙

임의의 역할의 특정 객체에 읽기연산은 반드시 하위 역할을 통해서만 가능하다.

이행 비밀성 보안 규칙의 경우는 이행 무결성 보안 규칙의 경우와 유사하게 정의될 수 있다. 즉, 일반적인 경우에는 역할이 가지고 있는 권한에 따라 특별한 제약없이 임의의 역할이 특정 객체에 읽기연산을 수행할 수 있으나, 이행 비밀성 보안 규칙은 그런 경우에 일어날 수 있는 보안 위반 사항의 발생을 방지하여 객체의 비밀성을 보장해 준다. 그림 4.8 [ii]의 이행 비밀성 보안 규칙은 다음과 같이 ECA 규칙을 이용하여 정의할 수 있다.

[ii]이행 비밀성 보안 규칙 ($\sim E_r$)

E:READ(R2$\rightarrow$O1)

C:$\sim$*DEFINED*(*READ*($R_2 \rightarrow O_1$)) $\wedge$ *DEFINED* (*READ*($R_1 \rightarrow O_1$) $\wedge$ *WRITE*($R_1 \rightarrow O_2$) $\wedge$ *READ* ($R_2 \rightarrow O_2$))

A:*PERMIT*((*READ*($R_1 \rightarrow O_1$), *WRITE*($R_1 \rightarrow O_2$), *READ*($R_2 \rightarrow O_2$))

이행 비밀성 보안 규칙은 역할 R_1이 객체 O_2에 읽기연산을 수행하고자 하는 사건(*READ*($R_2 \rightarrow O_1$))이 발

생한 경우에 적용된다. 일반적인 경우에는 역할 R_1이 객체 O_2에 읽기 접근 권한을 가지고 있기 때문에 특별한 규칙이 필요없이 읽기연산이 가능하지만, 이 경우에는 비밀성 유지를 위한 보안 규칙의 적용을 받아 역할 R_1이 객체 O_2에 직접 읽기연산을 수행하는 것이 불가능하다. 그 대신에, 반드시 역할 R_2를 통해서만 읽기연산이 가능하다. 즉, 먼저 역할 R_1이 객체 O_1에 대하여 읽기연산을 수행($READ(R_1 \rightarrow O_1)$하고, 역할 R_1이 객체 O_2에 그 정보를 기록한 다음($WRITE(R_1 \rightarrow O_2)$), 역할 R_2가 다시 객체 O_2로부터 그 정보를 읽으므로 써($READ(R_2 \rightarrow O_2)$) 정보의 비밀성을 보장하여 줄 수 있다.

상호배제 보안 규칙은 두 개 이상의 역할이 하나의 객체에 대하여 접근하려고 할 때, 이미 하나의 역할이 그 객체에 대하여 접근 연산을 수행하였다면 다른 역할들은 더 이상 그 객체에 대하여 접근을 할 수 없도록 하는 규칙이다. 상호배제 보안 규칙도 이행성 보안 규칙과 마찬가지로 보안 규칙 3과 보안 규칙 4로 나누어 정의할 수 있다.

다. 【보안 규칙 3】 상호배제 무결성 보안 규칙

임의의 역할이 특정 객체에 기록연산을 수행하면, 다른 역할들은 그 객체에 대한 기록 연산을 수행할 수 없다.

일반적인 경우, 다른 역할이 특정 객체에 기록연산을 수행하였는지의 여부와는 상관없이 그 객체에 대한 기록권한만 가지고 있다면 어느 역할이나 기록연산이 가능하다. 그러나 상호배제 무결성 보안 규칙은 이런 경우에 발생할 수 있는 정보의 불법적인 변경을 방지하므로 써 객체의 무결성을 유지하여 주는 보안 규칙이다. 그림 4.8 [iii]의 상호배제 무결성 보안 규칙을 ECA 규칙을 이용하여 표현하면 다음과 같다.

[iii] 상호배제 무결성 보안 규칙 ($E_{w1} \oplus E_{w1}$)

E: $WRITE(R_1 \rightarrow O_3)$
C: $OCCURRED(WRITE(R_2 \rightarrow O_2))$
A: $PREVENT(WRITE(R_1 \rightarrow O_3))$
또는
E: $WRITE(R_2 \rightarrow O_3)$
C: $OCCURRED(WRITE(R_1 \rightarrow O_3))$
A: $PREVENT(WRITE(R_2 \rightarrow O_3))$

위의 경우는 역할 R_1과 R_2가 객체 O_3에 기록연산을 수행하고자 하는 사건($WRITE(R_1 \rightarrow O_3)$, $WRITE(R_2 \rightarrow O_3)$)이 발생한 경우의 예이다. 일반적인 경우에는 역할 R_1과 R_2 모두 객체 O_3에 기록연산을 수행할 수 있다. 그러나 상호배제 무결성 보안 규칙의 적용을 받으면, 역할 R_1이 객체 O_3에 기록연산을 수행($WRITE(R_1 \rightarrow O_3)$)하고자 할 때(또는, 역할 R_2가 객체 O_3에 기록연산을 수행($WRITE(R_2 \rightarrow O_3)$)하고자 할 때), 이미 역할 R_2가 객체 O_3에 기록연산을 수행($WRITE(R_2 \rightarrow O_3)$)하였다면(역할 R_1이 객체 O_3에 기록연산을 수행($WRITE(R_1 \rightarrow O_3)$)하였다면), 역할 R_1의 객체 O_3에 대한 기록연산($WRITE(R_1 \rightarrow O_3)$)(역할 R2의 객체 O_3에 대한 기록연산($WRITE(R_2 \rightarrow O_3)$))은 수행이 금지된다.

라. 【보안 규칙 4】 상호배제 비밀성 보안 규칙

임의의 역할이 특정 객체에 읽기연산을 수행하면, 그 역할은 다른 객체에 대하여 읽기 연산을 수행할 수 없다.

보통의 경우에는 하나의 역할이 특정 객체에 읽기연산을 수행하였는지의 여부에 관계없이 다른 객체들에 대하여 읽기권한만 가지고 있다면 그 객체들에 대하여 읽기연산이 가능하다. 그러나 상호배제 비밀성 보안 규칙은 이런 경우에 발생할 수 있는 정보의 불법적인 유출을 방지하므로써 객체의 비밀성을 유지하여 주는 보안 규칙이다. 그림 4.8 [iv]의 상호배제 비밀성 보안 규칙은 ECA 규칙을 이용하여 다음과 같이 표현할 수 있다.

[iv] 상호배제 비밀성 보안 규칙 ($E_{r1} \oplus E_{r1}$)

E: $READ(R_3 \rightarrow O_1)$
C: $OCCURRED(READ(R_3 \rightarrow O_2))$
A: $PREVENT(READ(R_3 \rightarrow O_1))$
또는
E: $READ(R_3 \rightarrow O_2)$
C: $OCCURRED(READ(R_3 \rightarrow O_1))$
A: $PREVENT(READ(R_3 \rightarrow O_2))$

상호배제 비밀성 규칙은 역할 R_3가 객체 O_1 또는 O_2에 읽기연산을 수행하고자 하는 사건($READ(R_3 \rightarrow O_1)$, $READ(R_3 \rightarrow O_2)$)이 발생한 경우에 적용되는 보안 규

칙이다. 일반적인 경우에는 역할 R_3가 객체 O_1과 O_2에 읽기연산을 수행할 수 있다. 그러나 상호배제 비밀성 보안 규칙의 적용을 받으면, 역할 R_3가 객체 O_1에 읽기연산을 수행($READ(R_3 \rightarrow O_1)$)하고자 할 때(또는, 역할 R_3가 객체 O_2에 읽기연산을 수행($READ(R_3 \rightarrow O_2)$)하고자 할 때), 이미 역할 R_3가 객체 O_2에 읽기연산을 수행($READ(R_3 \rightarrow O_2)$)하였다면(역할 R_3가 객체 O_1에 읽기연산을 수행($READ(R_3 \rightarrow O_1)$)하였다면), 역할 R_3의 객체 O_1에 대한 읽기연산($READ(R_3 \rightarrow O_1)$)(역할 R_3의 객체 O_2에 대한 읽기연산($READ(R_3 \rightarrow O_2)$))은 수행이 금지된다.

임무분리 규칙은 역할들이 특정 객체에 접근하려고 할 때, 이미 이전에 그 객체에 접근한 적이 있다면 더 이상 그 객체에 대하여 접근할 수 없도록 하는 보안 규칙이다. 이 보안 규칙은 보안 규칙 5와 보안 규칙 6으로 다시 세분화될 수 있다.

마. 【보안 규칙 5】 임무분리 무결성 보안 규칙

임의의 역할이 특정 객체에 기록연산을 수행하면, 그 역할은 그 객체에 대하여 다시는 기록연산을 수행할 수 없다.

일반적인 경우, 임의의 역할이 특정 객체에 기록연산을 수행하였는지의 여부와는 상관없이 그 객체에 대한 기록 권한만 가지고 있다면 어느 경우에나 기록연산이 가능하다. 그러나 임무분리 무결성 보안 규칙은 이런 경우에 발생할 수 있는 정보의 불법적인 변경을 방지하기 위해 필요한 보안 규칙이다. 그림 4.8 [v]의 상호배제 무결성 보안 규칙을 ECA 규칙을 이용하여 표현하면 다음과 같다.

[v] 임무분리 무결성 보안 규칙 ($E_{w1} * E_{w1}$)
E: $WRITE(R_1 \rightarrow O_3)$
C: $OCCURRED(WRITE(R_1 \rightarrow O_3))$
A: $PREVENT(WRITE(R_1 \rightarrow O_3))$,
$PERMIT(WRITE(R_2 \rightarrow O_3))$
또는
E: $WRITE(R_2 \rightarrow O_3)$
C: $OCCURRED(WRITE(R_2 \rightarrow O_3))$
A: $PREVENT(WRITE(R_2 \rightarrow O_3))$,
$PERMIT(WRITE(R_1 \rightarrow O_3))$

위의 경우는 역할 R_1과 R_2가 객체 O_3에 기록연산을 수행하고자 하는 사건($WRITE(R_1 \rightarrow O_3)$, $WRITE(R_2 \rightarrow O_3)$)이 발생한 경우에 무결성을 유지하기 위한 보안제약 조건이다. 일반적인 경우에는 역할 R_1과 R_2모두 객체 O_3에 기록연산을 수행할 수 있다. 그러나 임무분리 무결성 보안 규칙의 적용을 받으면, 역할 R_1이 객체 O_3에 기록연산을 수행($WRITE(R_1 \rightarrow O_3)$)하고자 할 때(또는, 역할 R_2가 객체 O_3에 기록연산을 수행($WRITE(R_2 \rightarrow O_3)$)하고자 할 때), 이미 역할 R_1이 객체 O_3에 기록연산을 수행($WRITE(R_2 \rightarrow O_3)$)한 적이 있다면(역할 R_1이 객체 O_3에 기록연산을 수행($WRITE(R_1 \rightarrow O_3)$)한적이 있다면), 역할 R_1의 객체 O_3에 대한 기록연산($WRITE(R_1 \rightarrow O_3)$)(역할 R_2의 객체 O_3에 대한 기록연산($WRITE(R_2 \rightarrow O_3)$))은 수행이 금지된다.

바. 【보안 규칙 6】 임무분리 비밀성 보안 규칙

임의의 역할이 특정 객체에 읽기연산을 수행하면, 그 역할은 그 객체에 대하여 다시 읽기연산을 수행할 수 없다.

보통의 경우에는 하나의 역할이 특정 객체에 읽기연산을 수행하였는지의 여부에 관계없이 그 객체에 대하여 읽기 권한만 가지고 있다면 그 객체에 대하여 읽기연산이 가능하다. 그러나 임무분리 비밀성 보안 규칙은 이런 경우에 발생할 수 있는 정보의 불법적인 유출을 방지하기 위한 보안 규칙이다. 그림 4.8 [iv]의 임무분리 비밀성 보안 규칙은 ECA 규칙을 이용하여 다음과 같이 표현할 수 있다.

[vi] 임무분리 비밀성 보안 규칙 ($E_{r1} * E_{r1}$)
E: $READ(R_3 \rightarrow O_1)$
C: $OCCURRED(READ(R_3 \rightarrow O_1))$
A: $PREVENT(READ(R_3 \rightarrow O_1))$,
$PERMIT(READ(R_3 \rightarrow O_2))$
또는
E: $READ(R_3 \rightarrow O_2)$
C: $OCCURRED(READ(R_3 \rightarrow O_2))$
A: $PREVENT(READ(R_3 \rightarrow O_2))$,
$PERMIT(READ(R_3 \rightarrow O_1))$

임무분리 비밀성 규칙은 역할 R3가 객체 O1 또는

O2에 읽기연산을 수행하고자 하는 사건($READ(R_3 \rightarrow O_1)$, $READ(R_3 \rightarrow O_2)$)이 발생한 경우에 적용되는 보안 규칙이다. 일반적인 경우에는 역할 R_3가 객체 O_1과 O_2에 대하여 읽기권한만 가지고 있다면 아무런 제약이 없이 읽기연산을 수행할 수 있다. 그러나 임무분리 비밀성 보안 규칙의 적용을 받으면, 역할 R_3가 객체 O_1에 읽기연산을 수행($READ(R_3 \rightarrow O_2)$)하고자 할 때(또는, 역할 R_3가 객체 O_2에 읽기연산을 수행($READ(R_3 \rightarrow O_2)$)하고자 할 때), 이전에 이미 역할 R_3가 객체 O_1에 읽기연산을 수행($READ(R_3 \rightarrow O_1)$)하였다면(역할 R_3가 객체 O_2에 읽기연산을 수행($READ(R_3 \rightarrow O_2)$)하였다면), 역할 R_3의 객체 O_1에 대한 읽기연산($READ(R_3 \rightarrow O_1)$)(역할 R_3의 객체 O_2에 대한 읽기연산($READ(R_3 \rightarrow O_2)$))은 수행이 금지된다.

위에서 정의한 보안 규칙들은 일반적인 경우의 망관리 연산들과는 충돌이 되지만, 궁극적으로 정상적인 망관리 연산들의 수행을 보장하면서 보안 규칙들을 통해 정보의 비밀성과 무결성을 보장하므로써 시스템 전반의 보안을 유지해 주기 위하여 필요하다.

5. 결 론

망이 대규모화되고 복잡해짐에 따라 하나의 망관리 시스템에 의해 전체 망이 관리하기가 힘들어서 망관리는 여러개의 망관리 시스템에 관리행위가 분산되어 수행되어야 한다. 서로 다른 망관리 정책을 사용하는 각각의 망관리 시스템 사이에는 관리정보와 상호운용성을 통해 관리되어 진다.

본 논문에서는 각 망관리 시스템 사이에 관리객체에 대한 정보의 공유 및 상호운용을 위해서 하위역할의 접근권한이 상위역할에 상속되는 역할 격자구조를 각 망관리 시스템들이 보유하게하여 역할들의 권한부여 속성에따라 망관리 시스템 사이에 상호운용성을 제공하였다. 하나의 망관리 시스템의 역할 격자구조에 요구된 역할이 존재하지 않을 경우는 상호운용성을 제공하기 위해 역할 격자구조에 역할을 삽입하는 알고리즘과 하나의 격자구조에 또다른 격자구조를 병합하는 알고리즘을 제시하였다.

분산망 환경에서는 사용자와 자원이 널리 분산되어 있어 사용자와 자원사이에 복잡한 접근 제어 메카니즘이 요구된다. 즉 주체와 객체 사이에 정상적인 접근권한을 사용하여 접근 제어가 수행될 수 있지만 다른 주체와의 상호 관련성을 고려하여 접근권한을 관리함으로써 관리객체의 비밀성과 무결성을 보장할 수 있다. 즉 어떤 역할의 접근권한 고정되어 사용되지 않고 다른 역할과의 관련성에 따라 동적으로 변경될 수 있다.

본 논문에서는 역할 격자구조에 이행성, 상호배제와 임무분리 보안규칙 등의 보안규칙에 따라 하나의 역할과 다른 역할 사이의 상호 관련성을 고려하여 시스템의 상태에 따라 역할들의 관리권한을 동적으로 변경하여 비밀성과 무결성이 유지되도록 하였다. 마지막으로 역할의 동적인 변경을 위해 각각의 보안 규칙들을 ECA 규칙을 이용하여 표현하였다. 본 연구의 결과를 객체지향 개념을 기반으로 관리객체를 기술하는 OSI망이나 TMN망 등에 적용이 가능할 것으로 사료된다.

참 고 문 헌

[1] David d. Clark, David R. Wilson, "A Comparision of commercial and Military computer security policies," IEEE, 1987.

[2] Elke A. Rundensteiner, "Multiview:A Methodology for Supporting Multiple View in Object-Oriented Database," Proceeding of the 18th VLDB conference, pp187-198, 1992.

[3] F. H. Lochovsky, C. C. Woo, "Role-Based Security in Database Management Systems," Database Security:Status and Prospects, North-Holland, 1988.

[4] F. Rabitti, et al., "A Model of Authorization for Next Generation Database Systems," ACM Trans. on Database Systems, Vol. 16, No 1, March 1991.

[5] Ilsoo Ahn, "Database Issues in Telecommunications Network Mangement," ACM SIGMOD, May 1994, pp. 37-43.

[6] J. M. Veoni, "Overview of an Integrated Network Management Architecture for a Large Heterogeneous Network," NOMS '92 Vol. 1, 1992, pp. 279-289.

[7] Lmtiaz Mohammed and David M. Dilts, "Design for dynamic user-role-based security," Computer & Security, 1994.

[8] Masum Z. Hasan, "An Active Temporal Model for Network Management Database," Integrated Network Management IV, pp. 524-535, 1995.

[9] Mauro OLIVEIER, Nazim AGOULMINE, "Interoperability of Open Management Systems," Proceeding of the IFIP TC6/WG6.4 International Conference on Advanced Information Processing Techniques for LAN and MAN Management, pp. 209-224, 1993.

[10] Rabvi S. Sandhu, "Separation Of Duties In Computerised Information Systems," Database Security IV, 1991.

[11] Randy Chow and I-Lung Kao, "Modeling Complex Access Control Policies in Distributed Systems," 5th IEEE Computer Socity Workshop on Future Trend of Distributed Computing Systems, pp. 404-411, August 28-30, 1995.

[12] Ravi S. Sandhu, "Lattice-Based Access Control Models," COMPUTER, 1993.

[13] Ravi S. Sandhu, Pierangela Samarati, "Access Control:Principles and Practice," IEEE Communications Magazine, Sepetember 1994, pp. 40-48.

[14] Vicki E. Jones, Marianne Winslett, "Secure Database Interoperation Via Dynamic Addition of Roles," Security for Object-Oriented Systems, pp. 229-234, 1993.

[15] Wei, Wei, "On Managing Management System," Ph.D. dissertation, University of Missouri, Kansas City, U.S.A., 1993.

서 재 현

1985년 전남대학교 계산통계학과 졸업(이학사)
1988년 중앙대학교 대학원 전자계산학과(이학석사)
1996년 전남대학교 대학원 전산통계학과(이학박사)
1988년~1996년 송원전문대학 전자계산학과 전임강사
1996년~현재 목포대학교 컴퓨터공학과 전임강사
관심분야:통신망관리, 컴퓨터 네트워크 보안, 분산처리 시스템 등

김 태 연

1986년 전남대학교 계산통계학과 졸업(이학사)
1988년 전남대학교 대학원 계산통계학과(이학석사)
1996년 전남대학교 대학원 계산통계학과(이학박사)
1993년~1996년 광주예술전문대학 컴퓨터그래픽 디자인과 전임강사
1996년~현재 서남대학교 전산정보학과 전임강사
관심분야:통신망 관리, 분산처리 시스템, 통신 보안, 컴퓨터 그래픽스 등

노 봉 남

1978년 전남대학교 수학교육과 졸업(이학사)
1982년 한국과학기술원 전산학과(공학석사)
1994년 전북대학교 대학원 전산통계학과(이학박사)
1983년~현재 전남대학교 전산학과 교수
관심분야:객체지향 시스템, 통신망 관리, 정보 보안, 컴퓨터와 정보사회 등