역할기반 접근 제어를 적용한 데이터베이스 보안 시스템에서의 보안 정책 최소화

정민아* · 이광호*

Minimization of Security Policies in Database Security System applying Role-Based Access Control

Min-A Jung* · Kwang-Ho Lee*

요 약

데이터베이스 보안을 위해 주로 정책기반 접근 제어를 이용하며, 이 중 역할 기반 접근 정책의 도입은 강제적 접근 제어 정책과 임의적 접근 제어 정책의 단점을 보완하고자 하는 노력으로, 이를 통하여 적용 환경의 상황에 적합한 보안 정책을 유연하게 시행할 수 있다. 최근 사용자의 다양한 보안 요구사항을 수용하기 위해 정책기반 접근 제어 정책을 변형 및 적용하여 데이터베이스 보안 시스템을 구현한 바 있다. 이러한 시스템에서 보안 정책은 새로운 접근 제어가 필요할 경우 정책을 추가 생성하므로 같은 사용자에 따른 정책들의 중복 현상이 나타나 시스템의 성능을 저하시킬 수 있다. 본 논문에서는 기존의 접근 제어 정책을 변형 및 조합하여 적용한 데이터베이스 보안 시스템에서 역할 기반 접근 제어 정책을 적용하는 과정에서의 정책 중복의 문제를 소개하고 이를 해결하기 위한 정책 관리 모듈을 제안하고자 한다. 정책 관리 모듈은 사용자별로 생성되는 정책에 대하여 중복여부를 검사하고 중복된 정책에 대하여는 삭제하며, 사용자 별로 이미 생성된 정책에 데이터 그룹에 대한 접근 제어 정책을 통합할 수 있도록 구현하였다.

ABSTRACT

There are many security models for database systems using policy-based access control. RBAC (Role-based Access Control) is used for complementing MAC (Mandatory Access Control) and DAC (Discretionary Access Control) and is for performing flexibly security policies meet applied environment. We implemented the database security system that applies DAC, MAC, and RBAC to meet security requirements of users. However, security policies are constructed redundantly whenever security policies are needed to each user in this system. Even though the proposed security system can flexibly control more complicated 'read' access to various data sizes for individual users, it is obvious that there is a possibility that a new policy can be a duplication of existing policies. In this paper, we introduce the problem of policy duplication and propose the policy management module. With this proposed module, constructed policies are checked for duplication and deleted or merged with existing policies.

키워드

데이터베이스 보안, 역할 기반 접근 제어, 보안 정책 관리

* 목포대학교 컴퓨터교육과

접수일자 : 2005. 4. 26

Ⅰ. 서 론

데이터베이스에 대한 접근 제어는 운영체제에 의하여 관리되는 파일에 대한 접근 제어보다 더욱 복잡하다. 이는 데이터베이스 환경에서는 튜플, 애트리뷰트 등과 같이 파일보다 세밀한 객체에 대하여 접근 제어를 적용하여야 하기 때문이다[1]. 데이터베이스에서의 접근 제어는 데이터에 대한 허가 받지 않은 접근과 이용을 막기 위한 것이다. 데이터베이스 보안을 위해 적용되는 접근 제어 정책으로는 강제적 접근 제어(MAC: Mandatory Access Control), 임의적 접근 제어(DAC; Discretionary Access Control), 역할기반 접근 제어 (RBAC: Role-Based Access Control) 등이 있다. 이러한 접근 제어 정책은 각 정책의 특성을 고려하여 데이터베이스의 보안 요구 사항을 만족하기 위해 적용되어 왔다[2, 3, 4].

최근, 병원 등의 대규모 시스템에서는 환자의 질병 및 개인 정보와 같은 사적이고 기밀한 데이터를 관리하며, 이러한 시스템에 접근하는 사용자의 접근 요구 사항도 다양하다. 또한, 새로운 종류의 데이터가 빈번히 추가 및 변경되며, 이에 따라 데이터를 사용하는 사용자의 새로운 업무도 추가 및 변경이 필요하다. 이러한 보안 요구사항을 만족하기 위하여 접근 제어 정책을 변형 및 조합하여 다양한 유형의 접근 제어를 원하는 사용자의 보안 요구를 충족시키기 연구가 수행되었다[5,6]. 이러한 연구에서 데이터베이스를 접근하는 각 사용자별로 다양한 크기의 데이터 그룹에 대한 접근 제어를 제공하며, 임의의 정보에 대한 사용자의 접근 권한의 변화를 유연하게 수용한다. 이를 위해 강제적 접근 제어 정책, 임의적 접근 제어 정책, 역할기반 접근 제어 정책을 적절히 조합 및 변형하여 적용하였다. 데이터베이스 보안을 위한 역할 기반 접근 정책의 도입은 강제적 접근 제어 정책과 임의적 접근 제어 정책의 단점을 보완하고자 하는 노력이며, 이러한 정책의 도입으로 인한 장점은 여러 가지가 있다. 가장 두드러진 장점은 적용 환경의 상황에 적합한 보안 정책을 유연하게 시행할 수 있다는 것이다. 본 연구의 선행연구에서도 임의의 데이터에 대한 사용자의 접근 권한이 수시로 변동되는 상황에 융통성 있는 대응과 공통된 하나의 정보에 대하여 각 사용자 별 접근 제어를 제공하기 위하여 역할 기반 접근 제어 정책을 이용하였다[5]. 이러한 시스템을 위하여 데이터 그룹은 다양한 크기의 데이터 그룹으로 재정의하였고, 사용자 그룹은 보안등급에 의한 그룹, 역할에 의한 그룹, 사용자 부분집합으로 이루어진 특정 사용자 그룹으로 정의하였다. 특정 사용자 그룹에 대한 정책은 ***Block(s, d, r)***로 표현되며, 임의의 사용자가 다른 사용자에게 특정 데이터에 대한 권한을 임의적으로 허가할 수 없도록 하였다. 이러한 정책 설정을 가능하게 함으로써 임의의 사용자에 대하여 다양한 크기의 데이터 그룹에 대한 접근 제어를 수행할 수 있게 되었다. 그러나, 이러한 정책 설정은 새로운 접근 제어가 필요할 경우 정책을 추가 생성하므로 같은 사용자에 따른 정책들의 중복 현상이 나타나고, 정책들의 수가 증가함에 따라 시스템의 성능을 저하시킬 수 있다. 따라서 이와 같은 문제점을 해결하기 위해서는 정책을 생성하는데 있어서 정책이 중복되지 않고 정책의 추가, 삭제 등을 효율적으로 수행하기 위한 관리가 필요하다.

본 논문에서는 기존의 접근 제어 정책을 변형 및 조합하여 적용한 데이터베이스 보안 시스템에서 역할기반 접근 제어 정책을 적용하는 과정에서 나타나는 사용자별 정책 중복의 문제를 소개하고, 정책 중복 문제의 해결과 더불어 정책의 추가, 삭제를 효율적으로 수행하는 정책 관리 모듈을 제안하였다. 정책 관리 모듈을 통하여 사용자별로 생성되는 정책에 대하여 중복 여부를 검사하고 중복된 정책을 삭제할 수 있다. 또한, 새로운 정책의 생성이 요구될 경우 사용자 별로 이미 생성된 정책에 데이터 그룹에 대한 접근 제어 정책을 추가 또는 삭제할 수 있도록 구현하였다. 결과적으로 정책 관리 모듈은 보안 관리자가 보안 정책을 효율적으로 관리할 수 있도록 한다. 본 논문의 구성은 다음과 같다. 2장에서는 역할 기반 접근 제어 정책에 관하여 기술하고 3장에서는 본 논문의 대상인 데이터베이스 보안 시스템에서 생성되는 정책에 관하여 설명하고 4장에서는 제안한 정책 관리 모듈의 구조 및 기능에 관하여 설명한다. 마지막으로 5장에서는 결론을 맺는다.

Ⅱ. 관련연구

2.1 역할 기반 접근 제어

역할 기반 접근 제어 정책은 사용자가 적절한 역할에

할당되고 역할에 접근 권한(허가)이 할당된 경우에만 사용자가 특정한 모드로 정보에 접근할 수 있다. 즉 사용자-역할, 역할-허가에 대한 관계를 설정하여 추상적인 트랜잭션의 개념으로 접근을 제어한다. 이는 권한 관리를 매우 단순화시켜주고 기업의 특정한 보안정책을 구현하는데 있어서 유연성을 제공하며 사용자들은 업무적 권한과 책임에 따라 특정 역할의 구성원이 되서 접근구조의 변경 없이 역할의 변경을 쉽게 할 수 있다. 본 논문에서 사용한 계층적 역할기반 접근 제어 모델은 그림 1 에서 제시한 바와 같이 NIST에서 표준화한 RBAC 참조 모델이다[7,8].

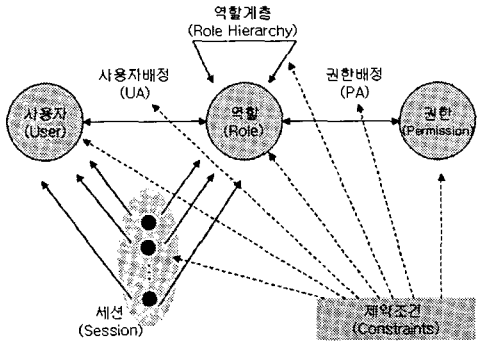

그림 1. 역할 기반 접근 제어
Fig. 1 Role-based access control

역할 기반 접근 제어를 사용함으로써 얻는 잇점은 몇 가지가 있다. 첫째, 사용자의 권한 지정을 두 부분으로 나누어 논리적으로 독립시킴으로써 보안 관리를 아주 단순하게 할 수 있다. 둘째, 역할에 대한 계층을 줌으로써 계층간에 상속을 할 수 있고, 이러한 계층적인 역할은 권한 부여에 대한 관리를 더욱 단순하게 할 수 있다. 셋째, 사용자에게 최소의 권한만을 허용함으로써 권한의 남용을 방지할 수 있다. 넷째, 임무 분리(Separation of Duty)를 함으로써 시스템 상에서 오용을 일으킬 정도의 충분한 특권이 사용된 사용자를 없게 한다. 다섯째, 사용자가 수행하는 활동에 따라 사용자를 분류할 수 있을 뿐 만 아니라 객체도 분류할 수 있다. 그러나 역할 기반 접근 통제의 유용성에도 불구하고 역할 기반 접근 통제 정책이 대규모 시스템에 적용될 경우 많은 역할과 역할 허가 사이의 복잡한 관계 설정으로 실제 시스템을 통제하기가 매우 어렵다[9].

Ⅲ. 데이터베이스 보안 시스템의 구조 및 기능

본 장에서는 본 논문의 선행 연구로 수행된 다양한 크기의 데이터 그룹에 대한 접근 제어를 지원하는 데이터베이스 보안 시스템에 관하여 기술한다[5].

3.1 시스템의 구조

보안 관리자는 사용자 그룹별로 테이블과 속성들에 대한 보안 등급과 역할에 따라 데이터를 접근하도록 하며, 부가적으로 다양한 크기의 데이터 그룹 DG_j에 따른 ***Block(s, d, r)***정책들을 생성 및 저장한다. 그림 2는 전체 시스템의 구조를 보인다.

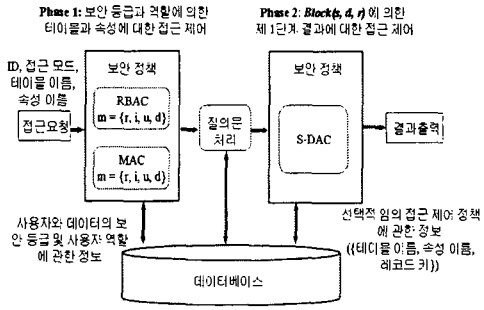

그림 2. 데이터베이스 보안 시스템 구조
Fig. 2 Database security system architecture

제 1단계에서는 먼저 접근하고자 하는 사용자 id를 검사하여 접근 요청이 SQL 문장 또는 역할에 의한 것인지를 파악한다. 만약, 접근 요청이 SQL 문장에 의할 경우 접근 모드, 테이블 이름, 속성들이 질의 문에서 추출되어 4장에서 기술한 정책1에 의해, 새로운 질의문이 생성된다. 또한, 역할에 의할 경우 사용자의 역할에 따른 접근 권한이 검사되고 할당된 접근 권한이 존재할 경우 RBAC 모듈이 수행된다. 접근 요청이 위의 정책들을 만족하지 않을 경우 접근 요청은 거부된다. 제 2단계의 변형된 DAC정책은 작은 수의 데이터 항목으로 구성된 다양한 크기의 데이터 그룹에 대하여 접근 제어한다는 의미에서 선택적 DAC(Selective-DAC:S-DAC)정책이라 정의한다. 이 단계에서는 1단계에서 수행된 결과가 ***Block(s, d, r)***에 의해 표현되는 정책에 따라 S-DAC에 의해 다양한 크기의 데이터 항목들이 선택적으로 필터링된다. 결과적으로 사용자는 위의 정책들을 모두 만족하는 데이터 항목만을 볼 수 있다.

3.2. 데이터 그룹에 관한 보안정책

위에서 제시한 ***Block(s, d, r)*** 에 의해 표현되는 정책은 특정 사용자가 다양한 크기의 특정 데이터 그룹에 대한 read를 허용하지 않도록 한다. 데이터 그룹에 대한 접근 정책은 ***Block(s, d, r)*** 로 표현되며, 다음과 같다.

> ***Block(s, d, r)***
> $s : S_j$, $j = \{1, 2, , u\}$, u 는 특정 사용자그룹의 수
> $d : DG_r$, $r = \{1, 2, , v\}$, v 는 데이터그룹의 수
> r : read mode

위에서 제시한 정책에서 데이터 그룹은 데이터는 테이블 이름(table name), 속성 이름(attribute name)과 레코드 키(record key)에 의하여 다양한 크기로 그룹화 되며, 이때 레코드 키는 원하는 레코드를 식별할 수 있는 포괄적인 정보를 의미한다. 이 방법에 의한 데이터 그룹은 다음과 같이 정의한다.

> $DG = \{ DG_1, DG_2, DG_3, DG_n \}$,
> n 은 데이터 그룹 수
> $DG_i = \{Table\ Name, Attribute\ Name(s), Record\ Key(s)\}$

위의 정의에 따라, 데이터 그룹은 다음과 같이 서로 다른 세 가지 유형을 갖는다.

> 테이블 전체: $DGi^1 = \{Table\ Name, NULL, NULL\}$
> 테이블과 속성: $DGi^2 = \{Table\ Name, Attribute\ Name(s), NULL\}$
> 테이블, 속성과 레코드 키: $DGi^3 = \{Table\ Name, Attribute\ Name(s), Record\ Key(s)\}$

Block(S_i, DG_q, r) 접근 제어 정책은 S_i 에 속한 사용자가 DG_q 의 데이터 그룹에 대하여 read 할 수 없음을 의미한다. 이와 같이 제2단계에서는 ***Block(s, d, r)*** 정책을 설정함으로써 사용자에 대하여 다양한 크기의 데이터 그룹에 대한 read 모드 접근 제어를 수행한다. 또한, 사용자에 대한 접근 제어 요구에 따라 ***Block(s, d, r)*** 정책을 설정할 수 있도록 하여 접근 제어 요구사항이 수시로 변하는 상황에 유연하게 대처할 수 있다.

IV. 보안 정책의 관리

Block(s, d, r) 정책은 특정 사용자에 대하여 다양한 크기의 데이터 그룹에 대한 접근을 막기 위해 생성된다. 그러나 ***Block(s, d, r)*** 정책은 새로운 접근 제어가 필요할 경우 계속 추가 생성되어 미리 생성되어 있는 정책의 데이터 그룹의 범위가 중복 또는 포함 관계에 있는 정책이 저장될 수 있다. 또한, 같은 사용자의 정책 중 데이터 그룹의 범위가 겹치는 부분을 갖는 경우 즉, 같은 속성 또는 레코드 키 등을 포함하고 있거나 서로 속성이나 레코드 키에 대하여 인접하는 데이터 그룹들에 대한 정책들도 저장될 수 있다. 이와 같은 경우에 사용자별로 생성되는 정책에 대하여 효율적인 정책 관리가 필요하다.

본 장에서는 3장에서 제시한 데이터베이스 보안 시스템에서 ***Block(s, d, r)*** 정책의 효율적인 관리를 위한 관리 모듈을 제안하고자 한다. 정책 관리 모듈은 새로운 정책을 생성할 경우 사용자별로 설정된 데이터그룹 단위에 관하여 이미 생성되어 있는 정책과의 중복을 줄이는 부분과 이미 생성되어 있는 정책과 통합하여 관리할 수 있도록 구성한다.

4.1 정책 관리 모듈의 구조

정책 관리 모듈은 중복된 정책을 제거하는 기능과 새롭게 생성되는 정책과 기존의 정책을 통합하는 기능을 지원한다. 이 모듈은 정책 최소화를 수행함으로써 시스템의 성능의 저하를 막기 위해 제안하였으며, 그림 3에서 제시한 바와 같이 S-DAC 모듈을 수행하기 전에 수행된다.

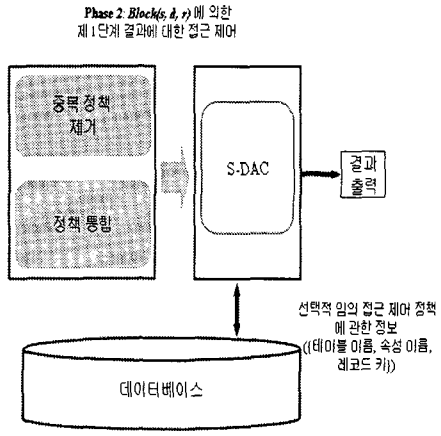

그림 3. 정책 관리 모듈
Fig. 3 Policy management module

4.2 중복된 정책 제거

중복된 정책을 제거한다는 의미는 각 정책에 정의된 데이터 그룹의 범위가 큰 데이터 그룹의 범위로 포함되

거나 같은 경우 크기가 작은 데이터 그룹에 대한 정책이 제거됨을 의미한다. ***Block(s, d, r)*** 정책에서 데이터 그룹은 다양한 크기이며, 그림 4에서 제시하였다. 데이터 그룹은 DG_I= *{Table Name, Attribute Name, Record Name }*로 표현되는 하나의 데이터 항목을 나타낸다.

테이블이름

$Attr_1$ (레코드키)	$Attr_2$	$Attr_3$	$Attr_4$	$Attr_5$	$Attr_6$
$Record_1$					
$Record_2$	**DG_1**				
$Record_3$					
$Record_4$			**DG_2**		
$Record_5$					

그림 4. 데이터 그룹
Fig. 4 Data group

그림 4의 데이터 항목은 다음과 같이 표현한다.

DG_1 = *{ Table Name, $Attr_2$, $Record_2$ }*
DG_2 = *{ Table Name, {$Attr_4$, $Attr_5$, $Attr_6$}, $Record_4$ }*

예를 들어, 사용자 S_1에 대한 보안정책이 그림 4에서 DG_1과 DG_2에 관한 정책이고, 추가로 다음과 같은 데이터 그룹에 대하여 정책을 생성한다고 가정한다.

DG_3 = *{ Table Name, {$Attr_4$, $Attr_5$, $Attr_6$}, { $Record_4$, $Record_5$ } }*
DG_4 = *{ Table Name, $Attr_2$, {$Record_2$, $Record_3$, $Record_4$ }}*

이 때, 사용자 S_1에 관한 보안정책은 다음과 같이 표현된다.

1. ***Block***(S_1, DG_1, ***r***),
2. ***Block***(S_1, DG_2, ***r***),
3. ***Block***(S_1, DG_3, ***r***),
4. ***Block***(S_1, DG_4, ***r***)

위에서 제시한 정책 중 1과 4, 정책 2와 3은 서로 데이터 그룹이 중복되는 정책이다. 그러므로 이러한 정책들은 서로 중복된 정책을 제거하고 크기가 큰 데이터 그룹에 대한 정책만을 유지함으로써 구성한다. 예를 들어, 정책 1 은 정책 4 와 중복되므로 이 중 한 정책을 제거하여 하나의 정책으로 유지한다. 즉, ***Block***(S_1, DG_1, ***r***)과 ***Block***(S_1, DG_4, ***r***) 의 정책은 작은 데이터 그룹에 대한 정책이 제거되어 새로운 정책인 ***Block***(S_1, DG_{14}, ***r***) 로 구성한다. 이 때 새로이 유지되는 데이터 그룹은 큰 데이터 그룹에 대한 정책인 ***Block***(S_1, DG_4, ***r***) 으로 저장되며, 결과적으로 중복된 정책의 제거는 두 정책 중 작은 데이터 그룹에 대한 정책인 ***Block***(S_1, DG_1, ***r***)가 제거되는 형태로 수행된다.

DG_{14} = *{ Table Name, {$Attr_2$}, { $Record_2$, $Record_3$, $Record_4$}}*

또한, 정책 2 와 정책 3도 같은 과정을 통해 작은 데이터 그룹에 대한 정책 2가 제거되고 새로운 정책 ***Block***(S_1, DG_{23}, ***r***)으로 정책 3이 유지된다.

이와 같이 같은 사용자에 대하여 데이터 그룹에 대한 제어가 중복될 경우 중복되는 부분을 제거함으로써 같은 사용자에 대한 정책 수가 감소하게 된다. 중복 정책 제거 알고리즘은 그림 5 와 같다.

```
DeletePolicy()
{
        for i = 1 - n - 1
          for t = i +1 - n
            INPUT: DGi = DGi^L,  DGt = DGt^R
            for k = 1 - p
              for l = 1 - q
                  if DGi == DGt
                  then DGi(k, l) = DGt(k, l),
                        DEL(match) = (k, l),
                        match++
                  else continue
               endFor
              endFor
            endFor
         endFor
        if match >= size(DGi)
                then delete DGi
}
```

그림 5. 중복된 정책 제거 알고리즘
Fig. 5 Redundant policy deletion algorithm

그림 5의 알고리즘은 같은 사용자의 정책 중 같은 테이블에 관한 정책들에 대하여 수행된다. 또한, 이를 위하여 데이터 그룹 DG_i의 Attribute Name(A_k)과 Record Key(R_l)를 행렬의 요소로 다음과 같이 표현하였다.

$DG_i = \{(A_1, R_1), (A_2, R_2), \ldots, (A_k, R_l)\}$,

$(1 \le k \le p, 1 \le l \le q)$이다.

단, DG_i^L은 비교행렬 요소를 갖는 DG_i이고,

DG_i^R은 대상행렬 요소를 갖는 DG_i이다.

4.3 정책 통합

보안을 위한 정책들은 사용자별로 정렬하여 유지함으로써 보안 관리자가 보안 관리를 하는데 있어서 편리함을 제공하게 된다. 사용자별 정책을 통합하기 위한 데이터 그룹들의 통합은 보안 관리자가 보안 지식을 바탕으로 할 수 있도록 지원한다. 이러한 정책 관리로 인하여 보안 관리자가 사용자별 정책을 확인할 수 있도록 하였으며, 정책에 대한 통합 여부는 전체 시스템의 보안 정책에 맞게 결정한다. 정책에 대한 통합이 필요하지 않을 경우는 기존의 정책에 단순히 추가하며, 통합이 필요하다고 판단될 경우 레코드 키 또는 속성을 기준으로 통합할 것인지를 결정한다.

예를 들어, 사용자 S_2에 대한 보안정책이 그림 4에서 DG_1과 DG_2에 관한 정책이고, 추가로 다음과 같은 데이터 그룹에 대하여 정책을 생성한다고 가정한다.

$DG_5 = \{$ Table Name, $Attr_4$, $Record_3$ $\}$

$DG_6 = \{$ Table Name, $\{Attr_5, Attr_6\}$ $Record_3$ $\}$

이 때, 보안 관리자는 데이터 그룹 DG_2, DG_5 와 DG_6은 서로 인접한 데이터 그룹들로 보안 관리자는 이러한 데이터 그룹들에 대한 통합 여부를 결정할 수 있다. 보안 관리자가 정책 통합을 원하지 않을 경우 사용자별 정책들은 그대로 저장되어 유지된다. 반면, 보안 관리자가 정책 통합을 원할 경우 속성을 기준으로 또는 레코드 키를 기준으로 통합여부를 물으며, 보안 관리자의 결정에 따라 위에서 제시한 세 개의 데이터 그룹 DG_2, DG_5 와 DG_6에 대한 정책들이 통합되어 새로운 정책인 $\boldsymbol{Block(S_2, DG_{256}, r)}$로 구성된다. 이 정책의 데이터 그룹은 다음과 같이 재정의하였다.

$DG_{256} = \{$ Table Name, $\{Attr_3, Attr_4, Attr_5, Attr_6\}$, $\{$ $Record_3$, $Record_4$ $\}\}$

정책 통합 알고리즘은 그림 6과 같다.

```
MergePolicies( )
{
    for i = 1 ~ n - 1
        for t = i +1 ~ n
          INPUT: DGi = DGi^L,  DGt = DGt^R
          for k = 1 ~ p
            for l = 1 ~ q
                if DGi == DGt
                then DGi(k, l) = DGt(k, l),
                     DEL(match) = DGt(k, l),
                     match++
                else continue
             endFor
            endFor
          endFor
        endFor
      if match != size(DGi)
        then {
          for w = 1 ~ match
           INPUT: DGi(k, l) = DEL(w);
}
```

그림 6. 정책 통합 알고리즘
Fig. 6 Policy integration algorithm

V. 결 론

다양한 크기의 데이터 그룹별로 접근 제어를 지원하기 위해 역할 기반 접근 제어를 적용한 데이터베이스 보안 시스템에서는 필요에 따라 보안 정책을 생성하기 때문에 각 사용자에 대한 정책의 데이터 그룹이 중복되는 등 정책의 수가 증가된다. 본 논문은 사용자에 대한 정책 수를 줄이기 위해 사용자 별로 중복되는 정책을 제거하고, 정책을 통합할 수 있는 있도록 하여 보안 관리자가 효율적으로 보안 정책을 관리할 수 있는 정책 관리 모듈을 제안하였다. 중복되는 정책에 대한 제거는 접근 제어 대상이 되는 데이터 그룹의 크기를 비교하여 작은 데이터 그룹에 대한 접근 정책을 제거함으로써 이루어진다. 또한, 보안 관리자가 사용자 별 정책들을 효율적으로 관리할 수 있도록 현재의 정책을 파악하고 통합 결정을 하는데 보다 쉬운 방법을 제시한다. 정책 통합은 보안 관리자에게 정책 통합 여부를 직접 결정하도록 결정권을 부여

하였으며, 보안 관리자가 정책 통합을 결정할 경우 데이터 그룹의 형태에 따라 속성 또는 레코드 키에 따라 통합을 결정하도록 하였다. 제안한 정책 통합 모듈은 통하여 ***Block(s, d, r)***의 정책 수를 감소함으로써 시스템의 성능 저하를 막을 수 있으며, 사용자별 정책들을 통합 관리함으로써 보안 관리자가 정책들을 쉽게 유지하고 관리할 수 있다.

참고문헌

[1] 김정덕, "데이터베이스 보안을 위한 통제기술," 데이터베이스 연구회 학회지, Vol. 16, No. 2, pp.3-6, 2000.

[2] M. Piattini and E. Fernandez-Medina, "Secure databases: state of the art," Security Technology, Proc. Of theIEEE 34th Annual 2000 International Carnahan Conference, pp. 228 237, 2000

[3] S. Lewis and S. Iseman, "Securing an object relational database," Computer Security Applications Conference, 1997.

[4] D. Ferraiolo, J. Barkley, and R. Kuhn, "A role-based access control model and reference implementation within a corporate intranet,"ACM Transactions on Information and Systems Security, Vol. 2, No.1, pp. 34-64, 1999.

[5] 정민아, 김정자, 원용관, 배석찬, "다양한 크기의 데이터 그룹에 대한 접근 제어를 지원하는 데이터베이스 보안 시스템," 한국정보처리학회논문지D, Vol. 10, No. 7, pp.1149-1154, 2003.

[6] R. Chandramouli, "A Framework for Multiple Authorization Types in a Healthcare Application System," 17-th Annual Computer security Applications Conference (ACSAC), Dec 10-14, 2001.

[7 D. Ferraiolo, R. Sandhu, S. Gavrila, D. Kuhn, and R. Chandramouli, "Proposed NIST Standard for Role-Based Access Control," ACM Transactions on Information and Systems Security, Vol. 4, No. 3, pp.224-274, Aug. 2001.

[8] R. Sandhu, E. Coyne, H. Feinstein, and C. Youman, "Role-based access control models," IEEE Computer, Vol.29, Issue 2, pp.38- 47, 1996.

[9] R. Sandhu and V. Bhamidipati, "The URA97 model for role-based user-role assignment," Database Security XI: Status and Prospects, Chapman and Hall, London, pp. 262-275, 1997.

저자약력

정민아(Min-A Jung)

1992년 전남대학교 전산통계학과 (학사)
1994년 전남대학교 대학원 전산통계학과(이학석사)
2002년 전남대학교 대학원 전산통계학과(이학박사)
2002년 ~ 2003년 광주과학기술원정보통신학과 Post-Doc.
2005년 ~ 현재 목포대학교 컴퓨터 교육과 교수
※ 관심분야 : 데이터베이스, 데이터마이닝, 생물정보학, 정보보호

이광호(Kwang-Ho Lee)

1987년 서울대학교 컴퓨터공학과 (학사)
1989년 한국과학기술원(공학석사)
1996년 한국과학기술원(공학박사)
1996년 ~ 현재 목포대학교 컴퓨터 교육과 교수
※ 관심분야 : 인공지능, 영상처리, 알고리즘