JKSCI

한국컴퓨터정보학회논문지
Journal of The Korea Society of Computer and Information
Vol. 27 No. 9, pp. 131-138, September 2022
https://doi.org/10.9708/jksci.2022.27.09.131

A Flexible Attribute-based RBAC Model

Si-Myeong Kim*, Sang-Hoon Han**

*Adjunct Professor, Department of Computer Science and Engineering, Dongguk University, Seoul, Korea
**Professor, Dept. of Computer Information Security, Korea National University of Welfare, PyeongTaek, Korea

[Abstract]

In this paper, we propose an FA-RBAC (FA-RBAC) model based on flexible properties. This model is assigned attribute-role-centric, making it easy to manage objects, as efficient as access control, and as the network environment changes, it can provide flexible access control. In addition, fine-grained permissions and simple access control can be achieved while balancing the advantages and disadvantages of the RBAC and ABAC models, reducing the number of access control rules by combining static attribute-based roles and dynamic attribute-based rules, and verifying the validity and performance benefits of the proposed model through comparison analysis and simulation.

▸ **Key words**: RBAC, access control, Attibute, Permission, FA-RBAC

[요 약]

이 논문에서 우리는 유연한 속성을 기반으로 하는 FA-RBAC(FA-RBAC)모델을 제안한다. 이 모델은 속성-역할 중심으로 할당하여 객체 관리가 용이하고 그만큼 접근제어의 효율성이 높으며 네트워크 환경이 변함에 따라 유연한 접근제어를 제공할 수 있다. 또한, RBAC 모델과 ABAC 모델의 장단점을 균형 있게 조정하면서 세분화된 권한과 간단한 액세스 제어가 가능하고 실현할 수 있고, 각 역할과 속성 간의 할당 관계를 정적 속성기반 역할과 동적 속성기반 규칙을 결합하여 접근제어 규칙의 수를 줄임으로써 시스템 오버헤드를 줄이고, 비교 분석 및 시뮬레이션을 통해 제안 모델의 타당성 및 성능 이점을 검증한다.

▸ **주제어**: 역할기반, 접근제어, 속성, 권한, FA-RBAC

• First Author: Si-Myeong Kim, Corresponding Author: Sang-Hoon Han
*Si-Myeong Kim (creta72@dongguk.edu), Department of Computer Science and Engineering, Dongguk University
**Sang-Hoon Han (shhan@knuw.ac.kr), Dept. of Computer Information Security, Korea National University of Welfare
• Received: 2022. 08. 18, Revised: 2022. 09. 19, Accepted: 2022. 09. 19.

http://www.ksci.re.kr pISSN:1598-849X | eISSN:2383-9945

I. Introduction

네트워크의 활용이 지속적으로 발달함에 따라 그 속도는 빨라지고 데이터 공유는 증가하고 있다. 이를 해결하기 위해 접근제어(access control)의 필요성이 나날이 높아지고 있다[1]. 접근제어의 유형으로는 강제적 접근제어(MAC)와 임의적 접근제어(DAC)가 대표적이다. 모든 시스템에서 사용자를 묶어 집단으로 접근제어관리를 한다. RBAC(Role Based Access Control)은 사용자 집단과 그에 따르는 권한을 이행하기 위해 역할을 정의하고, 그런 역할로 접근제어를 수행하는 가장 대표적인 모델이다[1][2].

또한, ABAC(Attribute-Based Access control)은 속성의 개념을 사용하여 주체와 객체의 속성에 대한 조건에 만족하는 속성을 정의하고, 속성별로 다른 주체를 식별하여 속성 간의 다양한 관계 여부를 검사하는 접근제어이다[5][6].

RBAC에 대한 연구는 지난 수십 년간 두 가지에 초점을 맞췄다. 첫째, 다양한 네트워크에서 기존 RBAC의 적응성을 향상 시켜 확장된 RBAC 연구와 특정 시스템에 적용하여 기업적 목표에 적용 확대하는 것이다. 그러나, 이러한 연구는 사용자-역할, 역할-권한에 대한 제한적인 요소를 본질적으로 반영하지 못했다. 따라서, 점점 더 복잡해지고 대규모로 분산되는 네트워크 환경에서 RBAC은 여전히 문제점을 보여준다. 그 문제점은 첫째, 사용자-역할 중심이기 때문에 할당된 객체를 관리하기 어렵고 그만큼 접근제어의 효율성이 낮다. 둘째, 네트워크 환경이 변함에 따라 유연한 접근제어를 제공할 수 없다. 예로, RBAC에서 n의 역할이 있는 경우, 접근제어 시스템에는 이를 구현하기 위해 n개의 속성이 필요하다. 속성이 존재하는 동안 2^n개의 역할을 사상한다. 또한, ABAC에서도 속성에 따른 역할은 2^n개에 이른다. 따라서, n이 증가함에 따라 접근제어를 위한 권한의 규칙의 수는 기하급수적으로 증가하기 때문에 시스템의 효율성이 떨어질 수밖에 없다.

이에 속성을 두 부분으로 구성한다. 하나는 RBAC의 역할과 유사한 ID, 이름, 직분 등과 같은 비교적 고정된 특성을 나타내는 역할과 할당된 정적 속성기반 역할 집합이고, 다른 하나는 시간, 요일, 직원 나이 등과 같이 접근제어 시스템에서 큰 변동이 있는 속을 나타내는 권한에 할당된 동적 속성기반 역할 규칙 집합이다.

본 논문에서는 속성이 있는 RBAC의 역할을 설명하고, 정적 속성기반 권한과 동적 속성기반 권한 규칙을 결합하여 역할의 수를 줄이고 접근제어 시스템의 부하를 줄이는 FA-RBAC을 구현한다. 본 논문은 1장에서 제안 모델의 관련된 연구와 필요성을 제시한다. 2장에서는 RBAC모델, Task-RBAC, Temporal-RBAC, ABAC의 각 모델을 살펴보고 3장에서는 모델 구현의 이론적 기초와 관련 정의 및 속성을 제시하고, 4장에서는 FA-RBAC모델을 제안하고, 기존의 모델과 비교 분석한다. 5장에서는 결론으로 끝을 맺는다.

II. Preliminaries

1. RBAC(Role Based Access Control)Model

RBAC은 1992년에 제안된 이후부터 지속적으로 발전한 모델이다[1]. RBAC의 기본 구조는 그림 1과 같이 Flat RBAC(RBAC0), Hierarchical RBAC(RBAC1), Constrained RBAC(RBAC2), Symmetric RBAC(RBAC3)으로 하위에서 상위 단계로 올라갈수록 하위 단계의 특성을 내포한다[1][2].

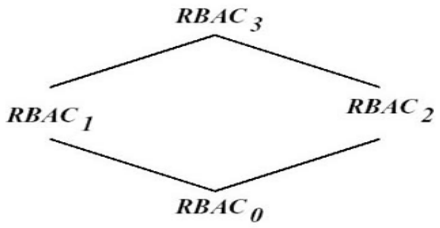

Fig. 1. Relationship among RBAC96 Model

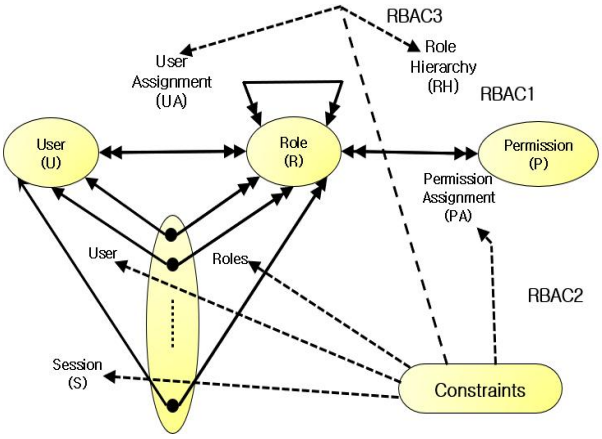

Fig. 2. RBAC96 Model

그림 2는 RBAC 구성 간의 관계를 나타낸 것이다. RBAC의 요소는 사용자, 역할, 권한으로 나누고, 사용자는 프로세서 또는 사람으로 볼 수 있고, 역할은 프로세서 또는 그 사용자에게 할당된 권한을 가진 조직 또는 시스템 내의 조직책임, 직무 이름, 프로세서이다. 권한은 권한을

할당한 시스템 또는 사용자에게 특정한 수행을 할 수 있는 수단이다.

RBAC0는 사용자-역할과 권한-역할으로 할당하여 다대다 관계를 나타낸다. RBAC1은 RBAC0(Flat RBAC)의 구성에 역할 계층(RH: Role-Hierarchical)이 추가된 것이다. 역할 계층은 조직 내에서 책임의 순서 또는 권한을 수행하기 위해 역할 계층적으로 구조화하는 것이다. RBAC2는 RBAC0에 통제를 두는 구조로 통제는 사용자-역할과 사용자-세션이 할당된 역할들의 활성화와 관련된다. RBAC3은 RBAC2에 권한-역할의 요구사항을 추가한 것이다[2][3][4]

그러나 RBAC모델은 다중 사용자, 다중 권한과 같이 세부화된 접근제어 환경에서는 효과적으로 접근제어를 할 수 없다는 단점이 있다. 이러한 전형적인 RBAC의 단점을 보안하기 위해 많은 연구가 이루어졌으며, 이들 중 대표적인 것이 ERBAC[8], Task-RBAC[11][12], Temporal-RBAC[13], OASIS RBAC[14], UCON[15], ABAC[4][9] 등 있다.

2. Task-RBAC

Task-RBAC(Task-Role-Based Access Control)은 RBAC모델에 작업의 개념을 도입하고, 사용자 1계층, 역할 1계층, 작업 1계층, 권한 1계층을 포함하는 4계층 제어모델을 설정한다.

Task-RBAC에서 운영 권한은 역할과 직접적인 관련이 없지만, 함수를 일련의 작업으로 나누고 특정 순서로 배열하여 작업에 할당한다. 그런 다음 관련 역할-작업을 할당하고 RBAC에서와 같이 사용자-역할 할당하고. 권한은 작업이 활성화된 경우에만 실행하여 이 작업을 완료하면 운영 권한이 사라지는 방식으로 권한에 따른 동적 접근제어를 수행한다.

Task-RBAC은 RBAC의 장점과 결합하여 사용자와 접근 권한을 논리적으로 분리하여 역할-작업을 할당한다. Task-RBAC의 구조는 그림 3과 같다.

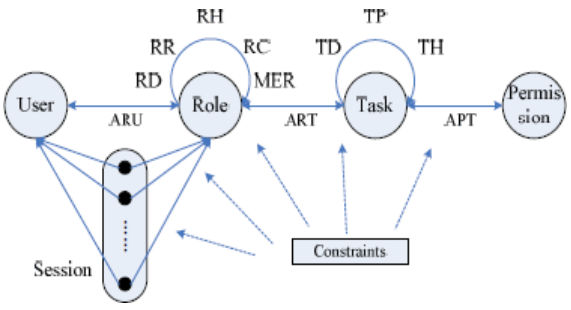

Fig. 3. Task-BAC(Task-Role Based Access Control) Model

3. Temporal RBAC

Temporal RBAC은 역할 활성화에 대한 시간적 제약을 지원할 수 있는 RBAC 모델의 확장이다.

이 기능에는 역할 트리거를 통해 표현되는 역할의 주기적 활성화/비활성화 및 시간적 의존성이 포함되며, 이는 역할 활성화 또는 비활성화에 기초하여 자동으로 실행되는 활성 규칙이다.

우선순위는 역할의 동시 활성화/비활성화가 필요한 경우 발생할 수 있는 충돌을 처리하기 위해 역할의 트리거 및 주기적 활성화/비활성화와 관련이 있다.

우선순위 및 우선순위 거부 규칙을 사용하여 충돌을 해결하여 관리자가 역할 활성화 및 비활성화에 대한 런타임 요청과 사용자에 의한 역할 활성화의 제한된 처리를 실행할 수 있도록 한다. 그러나 몇 가지 유용한 시간적 제약을 처리할 수 없다는 단점이 있다.

4. ABAC

ABAC(Attribute-Based Access Control)은 컴퓨팅 사용이 가능한 속성으로만 제한되는 접근제어(AC :Access control) 정책을 구현할 수 있는 유연한 접근 방식으로 분산 환경이나 빠르게 변화는 환경에 아주 이상적이다. 전형적인 AC는 직접 또는 미리 정의된 속성 유형을 통해 객체에 대한 작업 수행 기능의 실행을 요청하는 사용자의 ID를 기반으로 한다. 역할을 동적으로 변화시키는데 유연하지 않았고, 과도한 권한 할당도 복잡한 네트워크 환경에 적응하기에는 미흡했다. ABAC모델에서는 접근 권한은 주체가 제공하는 속성에 의해 결정되고, 접근 권한은 사용자가 접속이 시스템의 소거성 판별 기능을 만족하는 여부에 따라 달라진다. 또한, 속성 간의 관계에 따라 복잡한 권한 할당 및 접근제어 통제 조건이다.

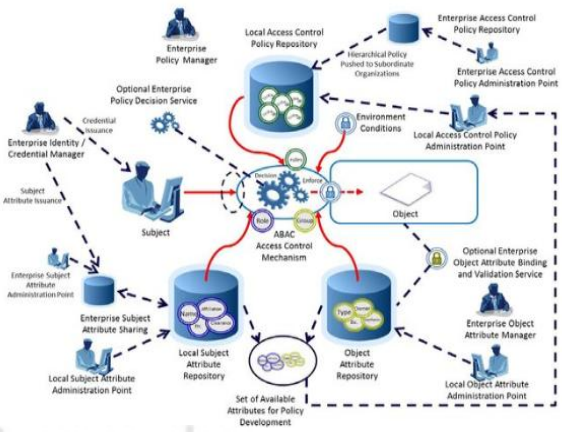

Fig. 4. ABAC(Attribute-Based Access Control) Model

그림 4과 같이 기본 정책, 속성 및 AC 메커니즘 요구사항 외에도 기업은 기업 정책 개발 및 배포, 기업 ID 및 주체 속성, 주체 속성 공유, 기업 개체 속성, 인증 및 AC에 대한 관리 기능을 지원해야 한다.

III. The Proposed Scheme

이런 관리 기능으로 최근 몇 년 동안 ABAC모델에 대한 이론적 연구는 대규모 분산 환경에서 요긴하게 사용됐다.

그러나 RBAC은 여전히 접근제어 모델의 주류이고 광범위한 적용 측면에서도 RBAC과 ABAC은 여전히 차이가 있다.

본 논문에서는 RBAC 모델과 ABAC 모델의 장단점을 균형 있게 조정하면서 세분화된 권한과 간단한 액세스 제어가 가능하고 실현할 수 있는 한 새로운 접근제어 모델인 FA-RBAC를 제안하고, 각 역할과 속성 간의 할당 관계를 정적 속성기반 역할과 동적 속성기반 규칙을 결합하여 접근제어 규칙의 수를 줄임으로써 시스템 오버헤드를 줄이고, 비교 분석 및 시뮬레이션을 통해 FA-RBAC 모델의 타당성 및 성능 이점을 검증한다.

1. Formal Definition of The Proposed model

본 논문에서 제안하는 FA-RBAC에서 속성은 역할의 특성화로 정적 속성(StA:Static-Attirbute)과 동적 속성(DyA:Dynamic-Attribute) 중심 역할로 구성된다.

[정의 1] 속성 (Attribute, A)

StA: 정적 속성 DyA: 동적 속성

Attribute ⊆ StA × DyA

[정의 2] 역할 (Roles, R)

R : 역할의 집합 OR: 객체 역할의 집합

SaR: StA에 의해 정의된 정적 속성 역할의 집합

DyR: DyA에 의해 정의된 동적 속성 역할의 집합

순서쌍을 가지는 역할의 요소이다.

[SaR, DyR]∈Roles

SaR1∪SaR2∪SaR3∪SaR4....∪SaRN∈ SaR

DyR1∪DyR2∪DyR3∪DyR4....∪DyRN∈ DyR

OR1∪OR2∪OR3∪OR4....∪ORN∈ OR

R ⊆ SaR × DyR × OR

[정의 3] 권한 (Permissions, P)

SaP: 정적 속성 역할에 할당된 권한

DyP: 동적 속성 역할에 할당된 권한

Pe: 객체에 할당된 권한

Pe = SaP ∩ DyP ∩ OP

∀pe⊆P, ∃SaP:SaP(SaP→SaR), ∃DyP:DyP(DyP→DyR)∩∀pe⊆P, ∃SaP:P(SaP∩SaR)∃DyP:P(DyP∩DyR)∈Pe ⇒ Pe⊆P

Pe = SaP∩DyP P ⊆ SaP×DyP

[정의 4] 역할-권한 할당 관계

PA : 권한 할당 관계

PA⊆ R×P:역할-권한에 대한 다대다 할당 관계

SaRPA⊆ SaR×SaP

:정적 역할-정적 권한에 대한 다대다 할당 관계

DyRPA⊆ DyR×DyP

:동적 역할-동적 권한에 대한 다대다 할당 관계

OPA⊆ OPS × SaRPA × DyRPA

:객체 역할-객체 권한에 대한 다대다 할당 관계

PA⊆ PSaRA×PDyRA×PORA

[정의 5] 사용자-역할에 대한 할당 관계

UA⊆ U×R : 사용자-역할에 대한 다대다 할당 관계

UStA⊆ U×SaR

:사용자-정적 역할에 다대다 대한 할당 관계

UDyA⊆ U×DyR

:사용자-동적 역할에 다대다 대한 할당 관계

UOA⊆ U×OR

:사용자-개체 역할에 다대다 대한 할당 관계

URA⊆ U×R

:사용자-역할 다대다 대한 할당 관계

위 정의에 의해 FA-RBAC에서 역할은 사용자와 권한을 연결하는 통로이면서 전형적인 RBAC모델과 마찬가지로 다양한 역할을 가질 수도 있지만, 하나의 역할은 여러 다은 사용들에게 속할 수도 있고 역할에는 여러 개의 서로 다른 권한이 있을 수도 있다. 또한, 모든 권한은 여러 역할에 의해 할당될 수도 있음을 보여주고 있다.

그림 5는 FA-RBAC모델에서의 사용자와 권한에 대한 할당 관계를 보여주고 있는 기본구조이다.

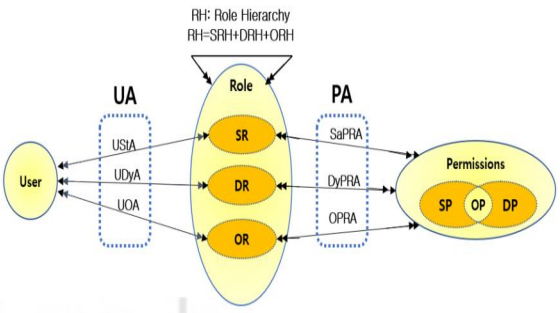

Fig. 5. The Relations of FA-RBAC

[정의 6] 역할 상속 (Role inheritance relation)

$\forall r1,r2 \in R \Rightarrow (r1,r2) \in RH = R \times R$

$\therefore r1 \trianglelefteq r2$: r1,r2는 속성이 순차적 부분 상속을 의미.

$\because$r1은 r2의 모든 권한을 얻을 수 있다.

r1,r1∈R, $\exists r1 \trianglelefteq r2$: R → a1은 r2의 속성이고, a2는 r1의 속성이다.

$\because \forall r1 \in R \cap a2 \subseteq a1 \Rightarrow r1 \trianglelefteq r2$

$\therefore \forall r1,r2 \in R, r1 \trianglelefteq r2 \Rightarrow a1=a2 \cap r1=r2$

$\forall r1,r2,r3 \in R, r1 \trianglelefteq r2 \trianglelefteq r3$: $R \Rightarrow a2 \subseteq a1 \cap a3 \subseteq a2$

$\therefore a3 \subseteq a1 \Rightarrow r1 \trianglelefteq r3$ 라는 것을 알 수 있다.

$SaRH \subseteq SaR \times SaR$, $\forall r \in SaR$, $\forall r1,r2 \in SaRH \subseteq SaR \times SaR$

$DyRH \subseteq DSaR \times DSaR$, $\forall r \in DyR$, $\forall r1,r2 \in DyRH \subseteq DyR \times DyR$

$RH \subseteq SaRH \times DyRH$, $\forall r \in R$, $\forall r1,r2 \in RH \subseteq SaRH \times DyRH$

$\therefore$ 여기서도 r1 $\trianglelefteq$ r2으로 상속됨을 알 수 있다.

[정의7] 제약 조건 (constraints)

제약 조건은 사용자, 역할 및 권한 할당에 제약을 두는 관계이다. 제약 조건을 충족하기 위해서는 시스템에 의해 개체가 허가되므로 모델 요소의 정당성이 보장된다. SaR에 일부 배타적 역할이 있고 다중 역할에 해당하는 동적 속성 규칙이 있는 경우 정적 역할은 상호 배타적입니다.

$\exists SaR1,SaR2 \in SaR \cap \exists DyR1 \in DyR \Rightarrow (U(si),(SaR1,DyR1)) \in UA(U(si),(DyR2,DyR1)) \in UA$, $SaR1 \cap SaR2 = \varnothing$

2. Model Structure

우리가 제안하는 FA-RBAC모델은 정적 속상 기반 역할과 동적 속성기반 규칙을 나타내는 속성 개념이다.

그림 6과 같이 Fa-RBAC모델의 구성 요소는 다음과 같다. 사용자(U:User), 역할(R:Role), 권한(P:Permission), 세션(S:sessions), 정적 역할(SaR:Static-Role), 동적 역할(DyR:Dynamic-Role), 정적 권한(SaP:Static-Permission), 동적 권한(DyP:Dynamic-Permission)으로 나타내고, 역할 계층(RH)에서 역할이 정적 속성 역할(SaR:Static-Attirbute role)과 동적 속성 역할(DyR:Dynamic-Attribute role)로 표현되는 (SaRH:Static Role hierarchy), DyRH:Dynaminc role hierarchy) 역할 계층을 제안한다.

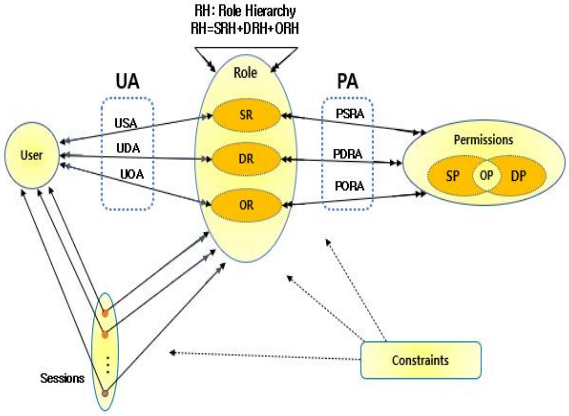

Fig. 6. The Proposed the FA-RBAC

3. Formal Definition of FA-RBAC

U:사용자집합 R:역할 P: 권한집합 S:세션집합

SaR : 정적 속성 역할 DyR: 동적 속성 역할

SaP : 정적 속성 권한 DyP: 동적 속성 권한

UStA : 사용자에 대한 정적 역할 사상 관계

UDyA : 사용자에 대한 동적 역할 사상 관계

UOA : 사용자에 대한 개체 역할 사상 관계

OPA : 객체 역할-권한 사상 관계

SaRPA: 정적 속성 역할-권한 사상 관계

DyRPA: 동적 속성 역할-권한 사상 관계

PA:권한에 대한 사상 관계

[정의 7] 본 구성원의 역할은 SaR, DyR로 구성되며 다대다의 관계이다.

-사용자 집합 : Users={u1,u2,u3.....un}

-역할 집합 : Roles $\subseteq$ SaR × DyR

-객체 집합: OBS={ab1,ab2,ab3....bn}

-연산 집합: OPS={op1,op2,op3....opn}

-세션 집합: Session={s1,s2,s3....sn}

-객체 권한 집합: OP=SaP∩DyP

-권한 집합: 모든 권한 집합은 P

-UAR⊆U×R, UStA⊆U×SaR, UDyA ⊆ U×DyR

-OPA⊆OPS×(SOR×DOR)

[정의 8] 각 사용자, 역할, 권한 요소들과 사상된 함수

-SeU(Si): Session → Users

한 사용자에게 각 세션이 사상된 함수

-SeR(Si): Session → 2^{role}

역할 집합이 하나의 세션 사상된 함수

단, (Role⊆SaR×DyR)∩(StAR(si)⊆{r|(StAU(Si),r)∈UA}

$\because$SeR(si)⊆{r∈R|(StAU(Si).r)∈UA}

-SSaR(si)⊆{SaR∈SaR|(StAU(Si).SaR)∈UStA}

정적 속성 역할의 집합-세션 사상 함수
-SDyR(si)⊆{SaR∈DyR|(StAU(Si).DyR)∈UStA}
동적 속성 역할 집합-세션 사상 함수
-US(u): User → 2session
세션의 집합-사용자 사상 함수
-assigned_p(r)= {p∈P|(p,r)∈PA}
r=(SaR,DyR)일 때, 권한 집합에 대한 역할 r의 사상 함수
-assigned_SaP(SaR)={SaP∈SaPerms|(SaP,SaR)∈SaPA}
정적 역할-정적 권한 집합에 대한 사상 함수
-assigned_DyP(DyR)={DyP∈DyP|(DyP,DyR)∈DyPA}
동적 역할-동적 권한 집합에 대한 사상 함수
-assigned_p(or)={p∈P|(p,or)∈OPA}
객체 역할과 권한 집합에 대한 사상 함수
-avail-session-SaP(s:Sessions) → 2^{SaP}
세션에 사용자가 수행할 수 있는 정적 속성 권한

단, $\sum_{r=0}^{s_i} assinged-saps(sar), r \in ssroles(s_i)$

-avail-session-DyP(s:Sessions) → 2^{DyP}
세션에 사용자가 수행할 수 있는 동적 속성 권한

단, $\sum_{r=0}^{s_i} assinged-dyp(dyr), r \in dyroles(s_i)$

-avail-session-p(s:Sessions) → 2^{P}
사상된 세션에 사용자가 수행할 수 있는 모든 권한
이 관계를 식으로 만들면 다음과 같다.

$$\sum_{r=0}^{s_i} assinged-p(r) = (\sum_{r=0}^{s_i} assinged-sap(sar)) \cap (\sum_{r=0}^{s_i} assinged-dyp(dyr)), r \in ssroles(s_i)$$

[정의9] 역할 상속이 발생할 때 UStA와 UDyA로 권한과 사용자 사상하는 함수 관계이다.

Permission*(u)={p:P|∃r∈SaR,(r,p)∈PSaRA∧(u,r)∈UStA}∨{p:P|∃r∈DyR,(r,p)∈PDyRA∧(u,r)∈UDyA}

4. Access Control Algorithm

다음은 본 논문에서 모델에 대한 설명을 기반으로, 사용자가 객체 리소스에 접근할 때에 따른 FA-RBAC모델의 접근제어 알고리즘이다.

Table 1. access control algorithm of the Proposed the FA-RBAC

input	u∈Users, ob∈OBS,op∈OPS
output	success or fail
1	s∈Sessions
2	if(s=Null) return fail
3	else r∈Role(s)
	SaR∈ SaRole(s),per_owned_SaR=assigned_SaP(SaR)
	DyR∈ DyRole(s),pem_owned_DyR=assigned_DyP(DyR)
	pem_owned=pem_owned_SaR&pem_owned_DyR perm_owned=assigned_pem(r)
	end if
4	if(pem_owned_r=pem_owned) per_allowed=assigned+pem(or)
5	if(pem_owned=pem_allowed) return success
6	else return fail

표 1과 같이 FA-RBAC모델에서는 먼저 인가된 사용자 U의 현재 세션을 Step 1에서처럼 추출한다. 그런 다음 Step 3에서 사용자 u의 세션에서 정적 속성 역할 할당과 동적 속성 역할 할당 모두 추출하는 세션의 역할에 대한 모든 권한을 얻는다. Step 4에서 표시한 접근 권한의 정확성을 확인할수 있고, 올바른 경우 접근제어는 사용자에게 해당 권한을 부여하고 성공(success)을 반환한다. 그렇지 않으면 요청을 거부하고 실패(fail)를 반환하는 알고리즘을 나타낸다.

5. The Proposed Model Analysis

다음 표 2는 다른 확장 RBAC 모델과 비교하여 제안하는 모델의 특징과 이점을 분석한다.

Table 2. A comparison of the proposed model and each model

	FA-RBAC Features	FA-RBAC advantages
RBAC	Roles are divided by attribute into static attribute-based roles and dynamic attribute-based role rules, and describe the attributes.	Enables fine-grained authorization and flexible configuration.
Task-RBAC Temporal-RBAC	Override assignment relationships between attributes and roles and describe different authentication relationships for users, roles, attributes, and permissions.	Enables dynamic management of roles and dynamic scaling of users.
Attibute-RBAC	Introducing dynamic property rules reduces the number of role rules required for access control.	Reduced access control system size and system consumption.

TRBAC ARBAC	Redefine the access control algorithm.	Simplify policy management.
Temporal-RBAC OAIS-RBAC	If DyR = {0} the FA-RBAC reverts to the existing RBAC model. That is, the RBAC model becomes a model in which dynamic attribute constraints of the FA-RBAC model are excluded.	Scalability as the number of users grows and scalability to other models.

제안 모델인 FA-RBAC에 대한 성능 분석을 위해 각 모델들과 초기화 시간을 비교 분석한다. 특히, 역할 테이블을 설정하는데 따른 시간 오버헤드를 각 모델들의 접근제어 정책과 제안 모델의 접근제어 정책을 비교하고, 역할 증가에 따른 모델 간의 실행 시간 변화를 통해 제안하는 모델인 FA-RBAC의 접근제어 알고리즘의 효율성을 입증한다.

실험 환경은 다음과 같다. 운영체제 win10pro, Intel(R) Core(TM)i7-7700HQ CPU@2.80GHz, 2.80GHz, DDR:32GB, VMware Workstation 16 Pro에 리눅스서버로 Fedora 36 server를 설치한다.

Visual studio 2022을 사용하여 역할-권한 및 역할 테이블을 프로그래밍하고 윈도우10 환경에서 접근제어의 효율성을 테스트하고, 그런 다음 각 모델과 비교하여 제안하는 모델의 접근제어 효율성에 대한 테스트를 한다.

각 모델은 역할 테이블을 생성하는 n개의 역할이 존재한다고 가정하고 기본 모델인 RBAC모델에 역할 생성에 따른 2^n-1 값을 따른다. 제안 모델의 경우 n개의 역할이 속성에 의해 먼저 x개의 정적속성 역할과 n-x개의 동적속성 역할로 분류된다고 가정하면 2^x+2^{n-x}-2의 역할을 생성하게 된다.

이 실험에서 최악의 경우는 동적 속성 역할의 수가 1로 제안 모델의 역할 생성 또한 RBAC모델과 마찬가지로 2^{n-1} 값 가지게 된다. 그림 7은 각 모델들과 제안모델의 초기화 효율을 비교한 것이다.

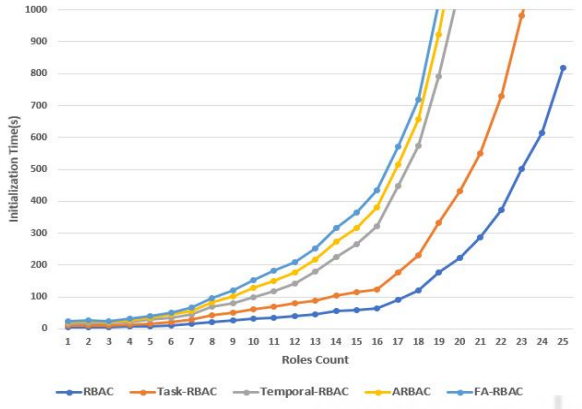

Fig. 7. An Efficiency Comparison of Initialization Time

그림 7에서 보는 바와 같이 역할의 수가 0~10인 경우 각 모델의 시간은 차이가 없다. 그러나 역할의 수가 12개 이상에 도달할 때, 각 시간적 이점은 FA-RBAC모델이 매우 낮은 초기화 시간을 나타내고 있다. 이것은 역할이 증가함에 따라 역할 분류에 따른 접근제어에서 그 역할 규칙의 수가 크게 줄었음을 의미한다. 이는 대규모 네트워크 시스템에서 제안한 모델이 여타 다른 모델들보다 초기화의 효율성이 상당히 높을 보여준다.

IV. Conclusions

본 논문에서는 기존 RBAC모델과 비교하여 제안한 FA-RBAC모델의 세분화된 권한 부여를 지원하고, 정책 관리를 단순화하며, 역할의 동적 확장을 지원하고, 확장성을 갖추는데 있어 유연한 모델이며, 운영 효율성이 높고 접근제어 시스템 소비량이 낮음을 보였다. 또한, 본 논문에서 제안 모델은 역할과 속성의 상응하는 관계를 보였고 역할과 정적 속성과 동적 속성 사이의 사상을 보였으며, 사용자, 역할 및 권한의 다중 인증 관계를 정의하고 마지막으로 제안 모델의 접근제어 알고리즘을 확인했다.

향후 연구과제로는 제안 모델을 더 대규모 접근제어 시스템에 적용할 예정이며, 모델의 알고리즘을 최적화하여 위임 모델에 적용하는 것을 연구할 것이며, 전형적인 위임 모델들과 비교하고자 한다.

REFERENCES

[1] Ravi S StAndhu, Edward j. Coyne, Hal L. Feinstein and Charles E. Youman," Role-based Access Control Model", IEEE, pp.38-47, Feb, 1996, DOI: 10.1109/2.485845

[2] Ezedin Barka and Ravi StAnhu, "Framework for Role-based Delegation Model and Some Extensions",Proceedings of the 23rd NIST-NCSC National Information Systems Security Conference, pp.101-114, Baltimore,UStA, October,2000, DOI: 10.1109/ACSAC.2000.898870

[3] Ravi Sandhu, " Role activation hierarchies" RBAC '98: Proceedings of the third ACM workshop on Role-based access controlOctober 1998 Pages 33–40, https://doi.org/10.1145/286884.286891

[4] Zhang L, Ahn .G.J and Chun B.T, "A Rule-based Framework for Role-based Delegation Revocation", ACM TranStActions on Information and System Security , Vol.6, No.3, pp404-441, August, 2003, https://doi.org/10.1145/373256.373289

[5] X. Zhang, Y. Li and D. Nalla, "An attribute-based access matrix

model", Proceedings of the 2005 ACM Symposium on Applied Computing, (2005), pp. 359-363. https://doi.org/10.1145/1066677.1066760

[6] Jiwan Ninglekhu, Ram Krishnan "AARBAC: Attribute-Based Administration of Role-Based Access Control", 2017 IEEE 3rd International Conference on Collaboration and Internet Computing (CIC), IEEE, 15-17 Oct. 2017, DOI: 10.1109/CIC.2017.00027

[7] Yan Xuexiong, Wang Qinxian, Xu Changzheng,"A Multiple Hierarchies RBAC Model",2010 International Conference on Communications and Mobile Computing,24 May 2010,pp 57-60, https://doi.org/10.1109/CMC.2010.117

[8] M. A. C. Dekker,J. G. Cederquist,J. Crampton,S. Etalle,"Extended privilege inheritance in RBAC", ASIACCS '07: Proceedings of the 2nd ACM symposium on Information, computer and communications security,March 2007, pp 383-385. https://doi.org/10.1145/1229285.1229335

[9] B. Malek and A. Miri, "Combining Attribute-Based and Access System", Proceedings of the 12th IEEE International Conference on Computational Science and Engineering, (2009), pp. 305-312. DOI: 10.1109/CSE.2009.157

[10] Tahmina Ahmed, Ravi StAndhu "Classifying and Comparing Attribute-Based and Relationship-Based Access Control" Conference: the Seventh ACM, March 2017. https://doi.org/10.1145/3029806.3029828

[11] S. Oh and S. Park, "Task-role-based Access Control Model", Information System, vol. 28, (2003), pp. 533-562. https://doi.org/10.1016/S0306-4379(02)00029-7

[12] Yingying Yu, Yan Chen, Yuqin Wen,"Task-role based access control model in logistics management system"Proceedings of 2013 IEEE International Conference on Service Operations and Logistics, and Informatics,26 September 2013,pp 130-135. DOI: 10.1109/SOLI.2013.6611396

[13] E. Bertino, P. Bonatti and E.Ferrari, "TRBAC: A Temporal Role-Based Access Control Model", ACM Transactions on Information and System Security, vol.4, no. 3, August 2001, pp. 191-223. https://doi.org/10.1145/501978.501979

[14] J. Park and R. Sandhu, "Towrads usage control models: beyond traditional access control", Proceeding of the 7th ACM Symposium on Access Control Models and Technologies (SACMAT' 02), Monterey, California, USA, (2002), pp. 57-64. https://doi.org/10.1145/507711.507722

[15] Matthew W Sanders,Chuan Yue, "Mining least privilege attribute based access control policies", JACSAC '19: Proceedings of the 35th Annual Computer Security Applications Conference December 2019, pp 404-416. https://doi.org/10.1145/3359789.3359805

[16] D. R. Kuhn, E. J. Coyne and T. R. Weil, "Adding attributes to role-based access control", Computer, vol. 6, (2010), pp. 79-81. DOI: 10.1109/MC.2010.155

[17] Bernhard J. Berger,Christian Maeder,RoDyRigue Wete Nguempnang, Karsten Sohr,Carlos Rubio-MeDyRano(Less) "Towards Effective Verification of Multi-Model Access Control Properties"Proceedings of the 24th ACM Symposium on Access Control Models and Technologies, pp 149-160, May 2019. https://doi.org/10.1145/3322431.3325105

Authors

Si-Myeong Kim received the B.S., M.S. and Ph.D. degrees in Computer Science and Engineering from Dongguk University, Secoul, Korea, in 2003, 2006 and 2011, respectively. Dr. Kim joined the faculty of the Department of Computer Science at Dongguk University, Seoul, Korea, in 2014. He is currently a Adjunct Professor in the Department of Computer Science and Engineering, Dongguk University and CEO of YK ENT co.,Ltd. He is interested in access control and delegation and Computing security and Artificial intelligence security.

Sang-Hoon Han received the B.S., M.S. and Ph.D. degrees in Computer Science and Engineering from Dongguk University, Korea, in 1990, 1995 and 2002, respectively. Dr. Han joined the faculty of the Department of Computer Information Security at Korea National University of Welfare, Pyeongtaek, Korea, in 2003. He is currently a Professor in the Department of Computer Information Security, Korea National University of Welfare. He is interested in Information Security, Internet of Thing(IoT) and Computer Vision, and multimedia computing.