역할기반접근통제를 이용한 연속적 접근통제

신욱[O*], 이동익*, 김형천**, 강정민**, 이진석**
*광주과학기술원 정보통신공학과
**국가보안기술연구소
*{sunihill[O], dilee}@kjist.ac.kr
**{khche, jmkang, jinslee}@etri.re.kr

The Continuous Access Control with RBAC

Shin, Wook[O*], Lee, Dong-Ik*, Kim, Hyoung-Chun**, Kang, Jung-Min**, and Lee, Jin-Seok**
*Dept. of Info. & Comm., Kwang-Ju Institute of Science and Technology
**National Security Research Institute

요 약

본 논문에서는 보안 운영체제를 구현함에 있어서, 보다 진보적인 형태의 접근통제를 시행하기 위한 새로운 접근통제 기법에 대하여 설명한다. 새로운 접근통제 기법은 역할 기반 접근통제 기법(RBAC)을 확장하여 구성한다. 또한, 개념의 정확한 표현 및 논리의 정확성 확인을 위하여 정형 기법을 이용, 접근통제 모델을 기술하고 분석한 결과에 대하여 설명한다.

1. 서 론

최근 컴퓨터 시스템 보안 문제의 심각성 및 이에 대한 원천적 해결 방안의 모색 필요성이 대두됨에 따라, 보안 운영체제에 관한 연구 개발이 활발히 진행되고 있다.

보안운영체제는 일반적인 운영체제 서비스에 인증, 암호화, 접근통제 등의 보안 서비스를 보강하여 구성된다. 이 중, 접근통제는 보안운영체제의 핵심 서비스가 되는 기술이며, 목적 환경의 보안정책에 큰 영향을 주는 중요한 서비스이다.

전통적으로 유닉스 호환 운영체제들은 임의접근통제(DAC: Disretionary Access Control)를 기반으로 운영되어 왔으나, 정보 흐름 통제의 불가로 인한 각종 보안 취약성이 지적됨에 따라, 강제접근통제(MAC: Mandatory Access Control)를 적용하여 보안성을 강화하려는 일련의 노력이 있어왔다[1,2,3]. 그러나, MAC의 지나친 제약은 이들 보안 운영체제들의 실용화에 제약을 가져왔으며, 최근에는 역할기반접근통제(RBAC:Role Based Access Control)를 적용하여 DAC과 MAC의 단점을 보완하고자 하는 노력이 있었다[4,5,6].

RBAC의 도입으로 인한 장점은 여러 가지이지만, 가장 두드러진 장점은 적용 환경의 상황에 적합한 보안정책을 일괄 시행할 수 있다는 것이다.

본 논문은 보안운영체제를 개발함에 있어서 RBAC 의 추상화 기법을 이용, 접근통제 과정에 절차적 제약을 도입하고자 하는 동기에서 출발한다.

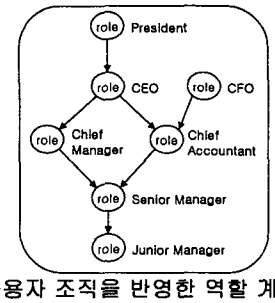
기존의 접근통제는 접근주체(사용자 혹은 시스템의 능동적 일부)가 접근객체(시스템 자원)에 접근하는 시점에서의 주, 객체 정보와 접근통제 규칙에 의거하여 접근의 적법성을 판단하는 과정이었다. 그러나, 이러한 접근통제는 TOCTTOU(time-of-check-to-time-of-use) 취약점과 같은 보안 정보 변조를 통한 시스템 침해 위협에는 대응할 수 없다. 물론, 이론적으로 이러한 변조에는 대응할 수 있는 방법들이 존재하지만, 시스템 구현 과정에서 포함된 버그나, 어플리케이션의 보안 취약점 등의 다양한 요인들이 취약점의 원인이 되므로, 현실적으로 보안 시스템의 완전성을 보장하는 것은 현실적으로 어렵다.

따라서, 접근통제의 범위를 좀 더 넓혀, 연속적인 접근 행위의 연관 정보를 접근통제 규칙에 반영해야 할 필요가 있다. 이러한 경우, TOCTTOU를 비롯한 다양한 형태의 공격에 보다 적절히 대응하는 진보적인 형태의 접근통제 시스템을 구현하는 것이 가능하다.

본 논문은 RBAC을 확장하여 이러한 공격들에 보다 적절하게 대처하는 접근통제 기법에 관하여 언급하고, 이를 정형적으로 모델화한 결과에 대해 설명한다. 논문은 다음과 같이 구성된다. 2장에서는 확장된 모델에 대하여 설명하고, 3장에서는 정형적으로 모델화 한 결과를 보이며, 4장에서 결론을 맺는다.

2. 확장된 RBAC 모델

RBAC의 주 구성요소는 사용자, 역할, 권한이다[7,8]. 기존 접근통제 방식과 구별되는 RBAC의 가장 큰 특징은 접근 주체와 객체 사이에 역할이라는 개체를 도입하여 주체와 객체의 추상화를 진행하고, 이를 통해 관리적 편의를 도모함에 있다. 역할은 접근 주체이면서 동시에 접근 객체로 해석될 수 있는 개체이다. 그러나, 여러 RBAC관련 연구들에 나타난 예제를 살펴보면, 역할 계층의 구성



[그림 1] 사용자 조직을 반영한 역할 계층의 구성 예

에 반영되는 주된 정보는 사용자 조직으로부터 추출되고 있음을 알 수 있다[7,8,9]. 그림 1은 사용자 조직을 반영하여 구성한 역할 계층의 전형이다[10]. Moffett[9]은 역할 계층 구성에서 사용되는 추상화 기법들을 집단화(aggregation), 일반화(generalization), 조직화(organization)로 분류하였으며, 그림 1에서 이러한 추상화의 결과를 확인할 수 있다.

이러한 특징을 가진 RBAC을 서론에서 언급한 바와 같은 형태의 접근통제 기법으로 확장하기 위해서는 추가적인 작업이 필요하다. 현재 RBAC의 접근객체, 즉 권한(permission)은 시스템 객체들과, 이에 대한 접근연산모드로부터 유도된다[12]. 예로, 사용자 암호를 담고 있는 파일 및 이에 대한 읽기, 쓰기, 수정 등의 연산이 권한에 대응된다. 물론, 권한들의 입상(granularity)은 접근통제 시스템의 적용 목적과 범위에 따라 달라질 수 있지만, 설명을 위해 RBAC의 권한 입상이 시스템콜 수준인 것으로 가정한다. 실제 보안운영체제들은 커널 함수 수준에 AEF(Access Enforcement Facility)[13]를 두고 있지만, 기본적으로는 운영체제의 제어 병목에 위치시키는 것이 통제에 가장 효과적이기 때문이다.

파일 접근 관련 권한들이 리눅스 시스템콜 sys_execve, sys_read, sys_write, sys_link, sys_unlink 등으로부터 파생된다고 하자. 기존의 접근통제 시스템은 각각의 시스템콜이 호출되는 순간, 호출자와 시스템 자원으로부터 접근통제 정보를 추출한 후, 접근통제 규칙에 위배되는지를 확인하고, 통제 해왔다. 그러나, 이들의 집합에 의미를 부여하여 보다 효과적인 접근통제를 실현할 수 있다. 예를 들어, TOCTTOU 취약점의 대표적 사례인 레이스 컨디션 공격[14]은 프로그램을 실행하는 도중 파일에 대한 링크와 링크 해제 동작을 반복하는 형태로 진행된다. 이러한 공격에 대한 대응 방안은 여러 가지가 있지만[15], sys_execve와 연관된 AEF의 호출 이후, sys_link, sys_unlink AEF 호출이 반복적으로 실행될 경우, 더 이상의 접근을 금지시켜 해결책을 마련할 수 있다.

이러한 통제 개념을 RBAC에 적용하려면, 권한들의 집합을 추상화 개체인 역할로 정의하고, 공격행위에 해당함을 명시할 수 있도록 본래의 개념을 확장하면 된다. 이러한 권한집합은 레이스 컨디션 공격을 의미하는 일종의 부정권한(*negative permission*)[16]으로 인식될 것이며, 접근통제는 권한집합의 수행 완료를 허용하지 않을 것이다. 이를 위해 원래 RBAC의 추상화 매체인 역할의 자료 구조 표현에 시스템 연산 실행 순서 표현을 위한 정보를 삽입해야 하며, 권한이 거부해야 할 것인지의 여부를 표기해야 한다. 이 밖에 연산 실행의 시각, 시간 정보, 객체의 소유자 등에 관한 정보 등을 추가하여 보다 정밀한 통제를 시행할 수 있다.

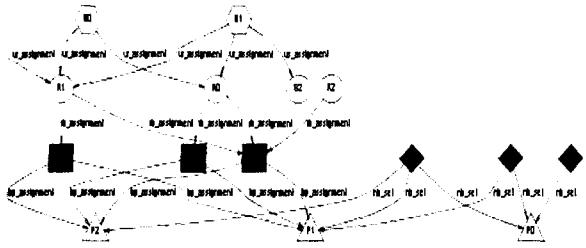
이렇게, 시스템 객체에 대한 정보를 추상화한 개체는, 사용자 조직에 관한 정보를 담고 있는 추상화 개체와는 의미와 표현이 달라진다. 이를 기존의 역할, 즉 사용자 추상과 같은 계층에 표현할 경우 다음과 같은 단점을 감수해야 한다.

- -구현상의 오버헤드: 사용자 조직을 표현하기 위한 자료구조에 연산 수행 순서, 시간, 소유권 등의 필드를 추가하게 되면 역할 개체관련 연산 구현 시 오버헤드가 된다.
- -의미적 괴리: 역할 이라는 같은 추상 계층 내에, 사용자 집합의 대표물과 시스템 자원의 대표물, 즉 의미가 다른 개체들의 혼재는 의미상 부적절하다.
- -규칙 생성 오류 위험: 주체로부터 객체로 이어지는 접근통제 개체간의 관계(relation)는 부분순서(partial order)를 가진다. 그러나, 단일 계층 내에 혼재된 주체 및 객체의 추상 개체는 보안 규칙 생성 시 객체가 주체를 상속받는 형태의 순서 역전 상태를 유도할 수 있으며, 이러한 순서 역전은 사이클을 생성하여 역할의 상속계층을 붕괴시킬 수 있다.

이러한 단점을 피하기 위하여, 사용자의 추상계층과 시스템 자원의 추상계층을 분리시키는 방안을 고려해 볼 수 있다. 기존 RBAC의 역할계층에서 사용자의 추상화만을 담당하고, 연산의 추상화를 담당하던 역할들을 새롭게 행위(behavior)라고 명명하고 역할과 별도로 정의하는 형태의 접근통제 기법을 가정해 보자. 확장된 기법의 주 요소는 사용자, 역할, 행위, 권한의 네가지로 구성된다. 역할은 접근 주체의 추상화를 행위는 객체의 추상화를 담당하는 계층이며, 주체의 속성과 객체의 속성에 따라 별도의 제약조건을 부여할 수 있다. 역할에 추가되는 제약조건으로는 기존 RBAC에서 제시하던 사용자 도메인, 접근 시간 등을 부여할 수 있으며, 객체의 제약조건으로는 연산수행 순서, 실행시간, 소유권 등을 부여할 수 있다.

확장된 접근통제 기법은 사실상 기존의 RBAC과는 구별되는 기법이다. 기존의 RBAC에서의 역할이란 접근주체로도, 객체로도 해석될 수 있는 중립적인 존재였으며, 주체와 객체간의 매핑(mapping)이 형상화된 개념이었다. 그러나 확장된 기법에서의 역할은 주체를 대표하는 개념, 행위는 객체를 대표하는 개념이며, 보안 규칙의 핵심이라 할 수 있는 주체와 객체간의 매핑은 역할과 행위와의 연관이며 개체(entity)화 되는 것은 아니기 때문이다. 시각을 달리하자면 역할은 그룹으로, 행위는 함수(function)으로도 해석될 수 있다.

확장된 기법을 기존 RBAC 모델과 같이 집합 수식에 기반하여 모델로 표현하는 과정은 한정된 지면으로 인하여 생략하고자 한다. 집합 수식에 기반한 모델 표현은 정형 모델 테스트에서 유추가 가능하다.



[그림 2] 확장된 RBAC의 정형 모델 도식화 결과

3. 확장된 RBAC의 정형 모델

이 장에서는 확장된 RBAC을 정형적으로 표현한 결과를 제시한다. 정형 표현은 Schaad[17]의 연구에서 사용된 바 있는 Alloy를 사용하였다. Alloy는 배우기 쉽고, 사용하기 쉬우면서도 구조적인 언어체계를 가진 정형언어이다. 또한 정형 표현 기술된 시스템의 마이크로 모델을 분석해주는 도구를 지원하고 있다. 그림 2는 정형적으로 기술된 확장된 RBAC 모델을 도식화 한 일부로서, 붉은 마름모로 표시된 실행 금지된 행위의 집합이 푸른 사각형으로 표시된 실행 허가된 행위와 동치되는 경우가 없도록 불변식(invariant)이 기술되어 있음을 확인할 수 있다.

4. 결론 및 향후 연구

본 논문에서는 보다 진보적인 접근통제의 실현을 위하여 유연한 접근통제 기법인 RBAC을 확장, 새로운 접근통제 기법을 제안하였으며, 정형적으로 모델링 하여 새 기법의 특성과 논리적 표현의 정확성을 확인하였다.

새롭게 제안한 접근통제 기법의 특징은 시스템 자원을 추상화 할 수 있는 독자적인 통로를 마련하는 것이며, 특히 추상화 과정에 절차적 제약을 도입하여 접근통제의 개념을 연속(continuous)된 연산 수행을 감시하도록 확장한 것에 있다. 절차적 제약은 부정적인 시스템 연산을 정의하고, 접근통제 시스템이 해킹 등 시스템 공격의 실행을 보다 정확히 구별하고 막을 수 있도록 한다. 이러한 기능은 침입탐지시스템(IDS)에서 볼 수 있는 기능과 유사한데, IDS가 어플리케이션 수준(level)에서 동작하는 반면, 접근통제는 우회할 수 없는 운영체제 커널 수준에서 동작하므로 제어에 유리하다.

절차적 제약을 자원 추상화 과정에 도입함으로써 얻을 수 있는 또 하나의 장점은 최소권한의 원칙을 보다 정확히 준수할 수 있다는 것이다. 기존의 권한 집합에는 절차의 개념이 존재하지 않으므로, 개개 권한을 어떻게 수행하든 허가하였으나, 새 기법에서는 지정된 순서에 따른 권한 실행만을 허용하는 것이 가능하다.

단, 3장에서 언급한 현재의 정형 모델은 절차를 모델화하지 않고 있다. 이는 나머지 모델의 기술이 정형 기법 중 증명 기반 기법을 이용하여 기술되었으나, 절차적 제약부분은 상태 기계 기반 기법을 이용해야 하기 때문이다. 따라서, 향후연구로 두 정형기법을 적절히 활용하여 정형 모델을 완성한 후, 실제 시스템에 구현하고자 한다.

5. 참고문헌

[1] "UNICOS Multilevel Security (MLS) Feature User's Guide," SG-2111 10.0, Cray Research, Inc. 1990

[2] M. Branstad, H. Tajalli, and F. Mayer, "Security issues of the Trusted Mach system," Proc. of 4th Aerospace Computer Security Applications Conference, pp. 362-367, 1998.

[3] Flask: http://www.cs.utah.edu/flux/fluke

[4] P. Loscocco, S. Smalley, "Integrating Flexible Support for Security Policies into the Linux Operating System," Proc. of the FREENIX Track: 2001 USENIX Annual Technical Conference (FREENIX '01)

[5] A. Ott, "The Rule Set Based Access Control (RSBAC) Linux Kernel Security Extension," 8th Int. Linux Kongress, Enschede 2001

[6] Trusted Solaris: http://wwws.sun.com/software/solaris/trustedsolaris/index.html

[7] R. Sandhu, E. Coyne, H. Feinstein, and C. Youman, "Role-Based Access Control Models," IEEE Computer, Vol. 29, No. 2 1996

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

[9] Moffett, J. D, "Control Principles and Role Hierarchies," 3rd ACM Workshop on Role Based Access Control (RBAC), George Mason University, Fairfax, VA, 22-23 October 1998.

[10] M. Koch, L.V. Mancini and F. P. Presicce, "A Graph-Based Formalism for RBAC," ACM Transactions on Information and System Security, Vol. 5, No. 3, pp. 332-365, August 2002.

[11] ITU-T SG/7 & Working Parties, "Final text for recommendation X.812 Information Technology-Open Systems interconnection Security framework for open systems: Access control framework," 1995.

[12]D. Ferraiolo, J. Cugini, R. Kuhn, "Role Based Access Control: Features and Motivations," Proceedings, Annual Computer Security Applications Conference, IEEE Computer Society Press, 1995.

[13] Linux Security Module: http://lsm.immunix.org/

[14] 8LGM, "Advisory 20", [8GM]-Advisory-20.UNIX.SunOS-sendmailV5.1-Aug-1995.
README

[15] M. Bishop and M. Dilger, "Checking for Race Conditions in File Access," Computing Systems 9(2) (Spring 1996), pp. 131-152.

[16] D. Gollmann, "Computer Security," John Wiley & SONS 1999.

[17] A. Schaad and J. Moffett, "A Lightweight Approach to Specification and Analysis of Role-based Access Control Extensions," 7th ACM Symposium on Access Control Models and Technologies (SACMAT 2002), Jun. 2002.