안전한 리눅스 시스템을 위한 E-BLP 보안 모델과 구현

강 정 민[†] · 신 욱[†] · 박 춘 구[†] · 이 동 익[††]

요 약

대부분의 안전한 운영체제는 주체와 객체에 보안 등급을 부여하여 운영하는 다중등급 정책(MLP : Multi-Level Policy)을 수용하고 있으며, BLP(Bell and LaPadula) 모델은 이 정책을 표현하는 검증된 대표적인 모델이다. 하지만 BLP 모델을 적용한 안전한 운영체제들은 사용자의 보안 등급을 프로세스에 그대로 상속하고 있음을 알 수 있다. 이러한 접근방법의 문제점은 프로세스를 전적으로 신뢰할 수 없다는 것에서 기인한다. 즉, 사용자의 보안 등급과 권한허용 범위를 오류가 내재되어 있거나 의도적으로 수정된 악의적인(malicious) 프로세스에게 그대로 상속할 경우, 시스템 안전성이 파괴될 가능성이 있다. 이는 BLP 모델이 접근 주체를 정의함에 있어서 시스템 사용자와 실제 그 접근을 대행하는 프로세스를 동일시하도록 단순하게 정의하고 있기 때문이며, 따라서 사용자와 프로세스간 신뢰관계를 모델에 도입함으로써 해결 가능하다. 또한 다중등급 보안 운영체제들은 접근 주체인 프로세스가 접근 객체로서 존재하는 등급화 된 프로그램 실행 시, 새로운 프로세스를 위한 보안 등급을 부여해야 하는데, 접근 주체와 접근 객체의 보안 등급이 다를 경우, 보안 등급 결정 문제가 발생하며 정보보호의 목적에 위배되는 결과가 발생한다. 이에 본 논문에서는 프로세스의 신뢰성을 고려하고, 보안 등급 결정 문제를 해결할 수 있는 확장된 BLP(E-BLP) 보안 모델을 제안하고 리눅스 커널(2.4.7)에 구현한다.

E-BLP Security Model for Secure Linux System and Its Implementation

Jung-Min Kang[†] · Wook Shin[†] · Chun-Gu Park[†] · Dong-Ik Lee[††]

ABSTRACT

To design and develop secure operating systems, the BLP (Bell-La Padula) model that represents the MLP (Multi-Level Policy) has been widely adopted. However, user's security level in the most developed systems based on the BLP model is inherited to a process that is actual subject on behalf of the user, regardless whatever the process behavior is. So, there could be information disclosure threat or modification threat by malicious or unreliable processes even though the user is authorized in the system. These problems can be solved by defining the subject as (user, process) ordered pair and by defining the process reliability. Moreover, when the leveled programs which exist as objects in a disk are executed by a process and have different level from the process level, the security level decision problem occurs. This paper presents an extended BLP (E-BLP) model in which process reliability is considered and solves the security level decision problem. And this model is implemented into the Linux kernel 2.4.7.

키워드 : BLP Model, E-BLP Model, Access Control, Reference Monitor, DRC(Dynamic Reliability Check)

1. 서 론

정보시스템에 대한 유·무형의 대규모 피해를 야기시키는 공격 시도들이 점차 에이전트화, 분산화, 자동화 및 은닉화의 형태를 띄고 있으며, 특히 리눅스 시스템의 해킹피해가 제일 큰 것으로 보도되고 있으며 앞으로도 증가할 것으로 예상된다[1]. 이런 공격들에 대한 정보시스템 보호의 목적은 보호해야 할 객체에 대해 인가된 사용자의 접근만을 허용하는 비밀성(Confidentiality), 인가된 사용자나 인가된 방식에 의해 객체의 수정을 허용하는 무결성(Integrity), 그리고 인가된 사용자에게 객체의 접근을 허용하는 가용성(Availability)을 보장하는 것이다[2]. 하지만 정보보호 기술로서 대두된 방화벽(Firewall)이나 침입 탐지 시스템(Intrusion Detection System) 같은 어플리케이션 수준에서의 정보보호의 노력은 안전한 운영체제의 기반이 없이는 실현될 수 없다[3].

대부분의 안전한 운영체제는 주체와 객체에 보안 등급을 부여하여 운영하는 다중등급 정책(MLP : Multi-Level Policy)을 수용하고 있으며, BLP(Bell and LaPadula) 모델은 [4-11] 이 정책을 표현하는 검증된 대표적인 모델이다. 하지만 BLP 모델을 적용한 안전한 운영체제들은 사용자의 보안

※ 본 연구는 한국 정보통신부 정보통신 연구센터 육성, 지원사업의 일환으로 수행되었습니다.
† 준 회 원 : 광주과학기술원 대학원 정보통신공학과
†† 종신회원 : 광주과학기술원 정보통신공학과 교수
논문접수 : 2001년 10월 8일, 심사완료 : 2001년 12월 19일

등급을 프로세스에 그대로 상속하고 있음을 알 수 있다. 이러한 접근방법의 문제점은 프로세스를 전적으로 신뢰할 수 없다는 것에서 기인한다. 즉, 사용자의 보안 등급과 권한허용범위를 오류가 내재되어 있거나 의도적으로 수정된 악의적인(malicious) 프로세스에게 그대로 상속할 경우, 시스템 안전성이 파괴될 가능성이 있다. 이는 BLP 모델이 접근 주체를 정의함에 있어서 시스템 사용자와 실제 그 접근을 대행하는 프로세스를 동일시하도록 단순하게 정의하고 있기 때문이며, 따라서 사용자와 프로세스간 신뢰관계를 모델에 도입함으로써 해결 가능하다. 또한 다중등급 보안 운영체제들은 접근 주체인 프로세스가 접근 객체로서 존재하는 등급화 된 프로그램을 실행할 때 새로운 프로세스를 위한 보안 등급을 부여해야 하는데, 접근 주체와 접근 객체의 보안 등급이 다를 경우, 보안 등급 결정 문제가 발생하며 정보보호의 목적에 위배되는 결과가 발생한다[12, 13]. 이에 본 논문에서는 프로세스의 신뢰성을 고려하고, 보안 등급 결정 문제를 해결할 수 있는 확장된 BLP(E-BLP) 보안모델을 제안하고 구현한다.

본 논문의 구성은 다음과 같다. 2장은 보안 운영체제가 제공해야 할 서비스인 접근 통제 서비스와 BLP 보안 모델에 대해서 기술하며, 3장에서 기존 BLP 모델에 기인한 문제점을 해결하는 개념적 E-BLP 모델에 대해 설명하고, 4장에서는 제안된 개념을 정형적인 E-BLP 모델에 표현한다. 5장에서는 본 연구팀이 개발중인E-BLP 모델을 적용한 CSRL시스템을 소개하고, 6장에서 결론 및 향후 연구 계획을 기술한다.

2. 관련 연구

이 장에서는 안전한 운영체제가 제공해야 할 접근 통제 서비스와 이를 구현하기 위해 일반적으로 채택되는 BLP 보안 모델에 대해 기술한다.

2.1 접근 통제 서비스[14-22]

접근통제 서비스란 참조 모니터(Reference Monitor)의 개념을 구현하여 접근 주체(Subject)의 접근 객체(Object)에 대한 요구를 보안 정책에 의거해 허가/거절하는 서비스이다. 접근 주체는 접근 객체의 사용을 요구하는 능동적(active)인 시스템 개체(entity)이며 주로 사용자 또는 프로세스가 이에 해당된다. 접근 객체는 정보를 저장하기 위한 수동적(passive)인 시스템 개체이며 주로 파일, 프로그램, 디렉토리, 디바이스 등이 해당된다. 이들 주체와 객체는 대부분의 경우 동적(Dynamic)으로 구분되고, 이들을 구분하는 정적(Static) 기준은 없다. 예를 들어 프로세스 간의 통신을 할 때 신호(signal)를 보내는 프로세스는 접근 주체가 되며, 신호를 받는 프로세스는 접근 객체가 된다. 다중등급 시스템에서 이들 주체와 객체들은 정보의 중요도(Sensitivity)에 따라 등급화가 되며, 부여된 보안 등급은 접근 결정을 위한 정보로서 사용된다. 접근 통제 시스템 즉, 안전한 운영체제를 개발하기 위해서 대부분의 연구사례 들은 BLP 보안 모델을 채택하고 있다[23-25].

2.2 BLP 보안 모델

BLP(Bell and LaPadula) 보안 모델은 정부의 기밀 분류 환경에서 상위 등급의 정보가 하위 등급으로의 흐름을 제한하는 다중등급 정책(MLP : Multi-Level Policy)을 표현하는 최초의 수학적 모델이다. 다음은 BLP 모델에서 시스템의 안전을 위해서 만족해야 하는 특성들이다.

- Simple Security Property(ss-property) : 주체의 보안등급이 객체의 보안등급을 지배(dominate)하면 해당 객체를 read할 수 있다.
- Star Property(*-property) : 객체의 보안등급이 주체의 보안등급을 지배하면 해당 객체에 write 할 수 있다.
- ds-property : 현재의 모든 접근은 접근행렬에 표시되어야만 한다. 즉 주체는 필요한 권한을 부여받은 접근만을 실행할 수 있다.

단지 여기에서 주(객)체의 보안등급이 객(주)체의 보안등급을 지배한다 함은BLP 모델이 중요한 정보의 보호에는 유용한 모델이나, 다음과 같은 문제점을 갖는다.

① 정보의 무결성(Integrity)을 배제하고 비밀성(Confidentiality)만을 고려한다. 다시 말해 악의적인 프로세스에 의해 정보가 불법적으로 변경(write)될 수 있는 가능성을 고려하지 않고 주체(사용자)에 의한 정보의 접근 통제만을 고려한다.
② 식별(Identification)과 인증(Authentication)과정을 거친 인가된 사용자가 악의적인 프로세스를 실행함으로서 중요한 정보에 접근(read)하여 정보의 비밀성을 파괴할 수 있다.
③ 접근 통제의 관리적 측면을 고려하지 않는다.
④ 공모에 의해 one bit information flow같은 covert channel이 존재할 수 있다.
⑤ Trusted subject를 요구한다.
⑥ 실세계에 적용하기에 너무 엄격하고, 불완전하여 추가적인 모델링을 요구한다.

문제점 ①, ②의 근본적인 원인은 사용자와 실제 시스템의 접근 주체인 프로세스를 동일시 함으로써 생긴다. 이제까지의 안전한 운영체제 연구 사례들은 BLP 모델을 적용함에 있어서 사용자의 보안등급을 프로세스에 그대로 상속함으로써 BLP 모델의 근본적인 문제점을 해결하지 못하고 있다. 본 논문에서는 ①, ②와 시스템의 가용성을 고려한다.

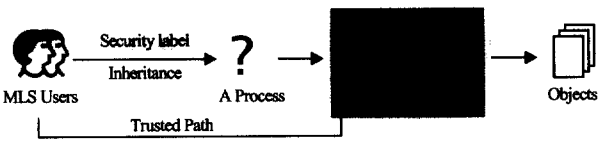
3. 개념적 E-BLP 모델

이 장에서는 시스템 내에서의 실질적인 접근 주체인 프로

세스의 안전한 행위를 보장하고, 실행 가능한 접근 객체의 실행 시 발생하는 보안 등급 결정 문제를 해결하기 위한 방안을 제시한다.

3.1 프로세스 신뢰성

(그림 1)은 BLP 모델기반의 시스템에서 주체의 객체에 대한 접근을 보여준다. 사용자는 ID/Password와 Level/Category 정보를 통해 식별 및 인증과정을 거친다. 인증 성공 시 사용자를 대신하는 프로세스가 수행 되며 그 프로세스는 시스템 내에서 실질적인 접근 주체가 된다. 하지만 시스템 내에서 사용자를 대신하는 프로세스의 정상적인 행위를 장담할 수는 없다. 다시 말해 참조 모니터는 프로세스의 등급 정보만을 가지고 접근 객체에 대한 접근통제를 수행함으로 비 신뢰적인 프로세스의 악의적인 접근 행위를 통제할 수 없다. 확장된 BLP(E-BLP)모델의 기본적인 아이디어는 시스템 내에서의 실질적인 접근 주체인 프로세스의 신뢰성을 보장하는 것이다[26, 27].



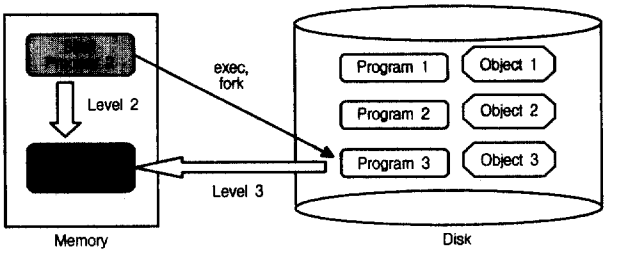
(그림 1) BLP 기반 시스템에서의 주체의 객체 접근

3.2 보안 등급 결정 문제

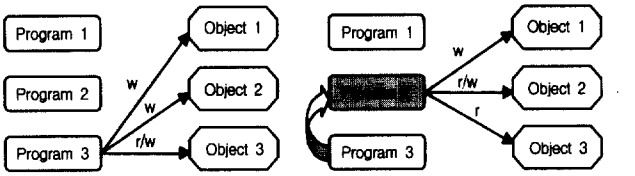
(그림 2)처럼 현재 실행되고 있는 2등급의 Process 2가 디스크상에 등급화된 접근 객체인 3등급의 Program 3을 실행(exec, fork 시스템 호출)한다고 가정하자. 물론 프로그램 실행 시 접근 통제(ss-property검사)에 의해 높은 등급의 주체는 낮은 등급의 객체를 read/execute할 수 있다. 하지만 이때 디스크상의 접근 객체의 보안 등급(3등급)과 접근 주체의 보안 등급(2등급)이 서로 다르다. 그렇다면 접근 객체인 Program 3이 접근 주체로서 메모리로 로드 되는 순간 접근 주체의 보안 등급과 접근 객체의 보안 등급 중 어떤 등급을 부여해야 하는지에 대한 보안 등급 결정 문제가 발생한다. 이 때 2등급의 주체(Process 2)는 Object 1에 대해서 write권한, Object 2에 대해서 read/write권한이 있고, Object 3에 대해서 read권한이 있다. 3등급의 접근 객체(Program 3)는 Object 1과 Object 2에 대해서 write권한이 있고, Object 3에 대해서 read/write권한이 있다.

3.2.1 접근 주체의 등급을 할당하는 경우

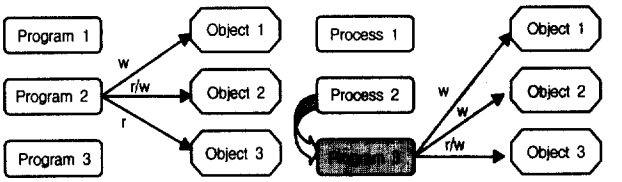
(그림 3)은 2등급의 접근 주체 Process 2가 3등급의 Program 3을 실행할 때, 실행된 Program 3이 접근 주체의 보안 등급인 2등급을 부여받는 경우이다. 원래 Program 3은 그림 왼쪽에서와 같이 BLP 모델의 규칙을 적용했을 때, Object 1과 Object 2에 대해서 write 권한이 있고, Object 3에 대해서는 read/write 권한이 있다. 하지만 Program 3은 실행 시 접근 주체인 Process 2의 보안 등급을 상속받음으로써 등급 향상의 결과를 가져온다. 그러므로 등급이 향상된 Program 3은 Object 2에 대해서 read가 가능해 졌으며, Object 3에 대해서는 write를 할 수가 없게 되었다. Object 2에 대한 read 허용은 정보의 비밀성(confidentiality) 문제를 일으킬 수 있으며, Object 3에 대한 write권한이 허용되지 않는 것은 가용성(usability)을 낮추는 결과이다.



(그림 2) 접근 객체 실행 시 보안 등급 결정



(그림 3) 접근주체의 등급을 할당하는 경우



(그림 4) 접근객체의 등급을 할당하는 경우

3.2.2 접근 객체의 등급을 할당하는 경우

(그림 4)는 2등급의 접근 주체 Process 2가 3등급인 Program 3을 실행할 때, 실행된 Program 3이 자신의 원래 보안 등급으로 실행되는 경우이다. 원래 접근 주체인 Process 2는 그림 왼쪽에서와 같이 BLP 모델의 규칙을 적용했을 때, Object 1에 대해서 write권한, Object 2에 대해서 read/write권한이 있고, Object 3에 대해서는 read 권한이 있다. 하지만 Program 3이 적재되어 Process 3으로 실행 시, 원래 접근 주체인 Process 2의 보안 등급은 2등급 그대로이나, 새로 생성된 Process 3의 보안등급은 객체일 당시의 보안 등급인 3등급을 상속받음으로써 동일 사용자에 의한 일련의 작업에 있어서 대행 주체인 Process들의 보안 등급이 일치하지 않음으로써 정보의 무결성(integrity)을 저해할 수 있다.

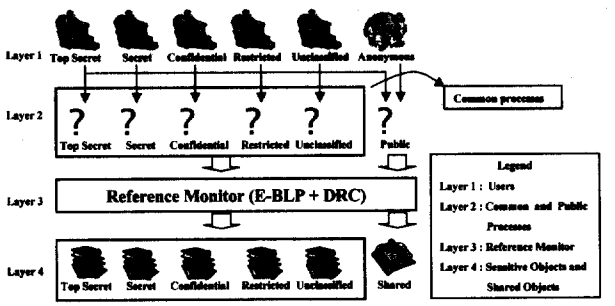
3.2.3 보안 등급 결정 문제 해결

등급 결정 문제를 피하기 위해 실행가능 한 접근 객체들을

공통의 영역(본 논문에서는 Common 영역)으로 묶고 이 영역의 접근 객체들은 보안 등급화가 되어 있지 않다가 접근 객체들을 실행할 때 접근주체의 보안 등급으로 상속하는 방법을 택한다. 이는 로그인 시 사용자의 의도로 제시된 보안 등급을 시스템내의 접근 통제 모듈에 의해 계속적으로 사용되게 하기 위해서이다. 그러므로 등급 결정 문제 자체를 피할 수 있고, 등급 결정 문제에서 생기는 비밀성, 무결성, 가용성의 문제는 발생하지 않는다. 하지만 일부 실행 가능한 접근 객체들이 실행시 반드시 무결성을 보장할 수 있는 것은 아니다. 다시 말해 악의적이거나 취약하다고 알려진 프로그램들은 보호해야 할 시스템 객체들(파일, 디렉터리 등)을 접근 시 원하지 않는 행위를 할 수 있으므로 분류(본 논문에서는 Public영역)되어야 한다. 또한 Public 영역의 접근 주체들은 보안등급이 부여되지 않은 익명의 사용자들을 위해 존재한다. 이는 시스템의 가용성을 증대시키며, 접근 객체의 대상을 공유 객체들로(Shared Object) 제한시킴으로 등급화 된 접근 객체들의 기밀성, 무결성을 보장하기 위함이다. <표 1>은 실행 가능한 접근 객체들의 분류 예를 보여준다.

〈표 1〉 실행 가능한 접근 객체들의 분류 예

	Common	Public
System Programs	Shell Scripts, Utilities, Commands	Daemons(httpd, ftpd)
User Programs	Applications, Programs (Editor, Office PGs)	User developed Programs, Hacking PGs.



(그림 5) 분류된 프로그램들의 접근 진행

(그림 5)는 위에서 분류된 프로그램들의 실행 시 접근 흐름을 보여준다. Common 영역의 프로그램들은 접근통제(Reference Monitor)에 의해 Layer 4의 등급화 된 객체들을 접근할 수 있으며, Public 영역의 프로그램들은 공유 객체들로 접근이 국한된다. 그리고 두 영역의 접근 주체들은 접근통제 모듈인 Layer 3에서 임의의 프로세스의 악의적인 행위 및 예상치 못한 행위를 판단할 수 있는 ***DRC(Dynamic Reliability Check)***에 의해 무결성 검사가 이루어진다. 또한 DRC는 보안 관리자에 의해 실행 가능한 접근 객체들을 분류하기 위한 도구로도 사용이 된다. 4.2에는 동적인 프로세스의 신뢰성을 검사하는 DRC의 기능을 ***fr(s)*** 함수로서 모델에 표현했다.

4. 정형적 E-BLP 보안 모델

이 장에서는 3장에서 설명된 개념들을 적용하는 확장된 BLP 보안모델의 구성 요소들과 특성 함수들을 정형적으로 기술한다.

4.1 구성요소

4.1.1 사용자 : U

- U = User Levels P(Categories).
  단, P(x)는 임의의 집합 x에 대한 멱집합(power set)을 나타낸다.
- User Levels = {Top Secret, Secret, Confidential, Classified, Unclassified, ***Anonymous***}
- Categories = {dept A, dept B, dept C, etc}

Anonymous 레벨은 시스템내의 등급화 된 사용자가 아닌 외부의 사용자들을 위해 존재하며, 공유(***Shared***) 객체만을 접근하도록 제한된다.

4.1.2 프로그램 : P

- $P = Common \cup Public$

프로그램들은 신뢰적인 프로그램 영역인 Common과 공유 객체로의 접근만을 허용하는 Public영역으로 나뉜다. 또한 두 영역의 프로그램들은 신뢰성 있는 경로를 보장하기 위해 참조 모니터에 의해 제어된다.

4.1.3 접근 주체 : S

- $S = U \times P$
- U : 등급화된 사용자들의 집합
- P : 프로세스들의 집합

사용자와 접근 통제시 사용자를 대신하는 동적인 주체가 되는 프로세스를 주체로서 정의한다. 이는 동적인 프로세스의 행위를 고려하기 위함이다.

4.1.4 접근 객체 : O

- O = Object Levels P(Categories).
  단, P(x)는 임의의 집합 x에 대한 멱집합(power set)을 나타낸다.
- Object Levels = {Top Secret, Secret, Confidential, Classified, Unclassified, ***Shared***}
- Categories = {dept A, dept B, dept C, etc}

시스템의 가용성과 중요한(sensitive) 객체들(Top Secret ~Unclassified)의 보호를 위해 Anonymous 등급 사용자의 접근은 ***Shared***객체로 제한된다.

4.1.5 접근 동작 : **A**

- A = {r, w, a, e}
- r : 객체에 포함된 정보를 읽기.
- w : 정보의 내용을 읽고 쓰기.
- a : 객체 정보를 읽을 수 없고, 새로운 정보 추가.
- e : 실행 가능 접근 객체를 실행.

4.2 시스템 상태 : **V**

- V = (B, M, F) : 시스템의 상태를 기술한다.
- B = (S, O, A) : 접근 주체 ***S***가 접근 객체 O를 A의 접근 동작으로 접근하는 것을 의미한다.
- M : 접근 주체가 접근 객체에 대해 허가된 동작을 명시하는 접근행렬로서 행은 접근 주체를, 열은 접근 객체를 나타낸다. 단 M의 각 요소는 M[s, o]로 표시한다.
- F = (fu, fc, fo, ***fp, fr***) : 주체/객체의 등급 검사 함수들, 프로그램 영역 검사 함수 및 동적 프로세스 신뢰성(무결성) 검사 함수(DRC)이다.
- fu : S → User Levels는 사용자의 최고 보안등급을 검사하는 함수이다.
- fc : S → User Levels는 사용자의 현재 로그인 등급을 검사하는 함수이다.
  단, ∀s ∈ S, fu(s) ≥ fc(s). (≥ : dominance relation)
- fo : O → Object Levels는 접근 객체의 보안등급을 검사하는 함수이다.
- ***fp : S → P는 실행중인 프로세스의 영역을 검사하는 함수이다.***
- ***fr : S → Boolean는 실행중인 프로세스의 신뢰성을 검사하는 함수이다.***

4.3 보안 규칙

시스템이 안전하기 위해서 만족해야 할 확장된 특성들이다.

4.3.1 **e-ss-property**(Extended Simple Security Property)

- if M[s, o] = r, ***(fp(s) = common)*** ∧ ***(fr(s) = true)*** ∧(fc(s) ≥ fo(o)).

4.3.2 **e-*-property**(Extended Star-Property)

- if M[s, o] = a, ***(fp(s) = common)*** ∧ ***(fr(s) = true)*** ∧(fc(s) ≤ fo(o)).
- if M[s, o] = w, ***(fp(s) = common)*** ∧ ***(fr(s) = true)*** ∧(fc(s) = fo(o)).
- if M[s, o] = r, ***(fp(s) = common)*** ∧ ***(fr(s) = true)*** ∧(fc(s) ≥ fo(o)).

4.3.3 **e-ds-property**(Extended ds-Property)

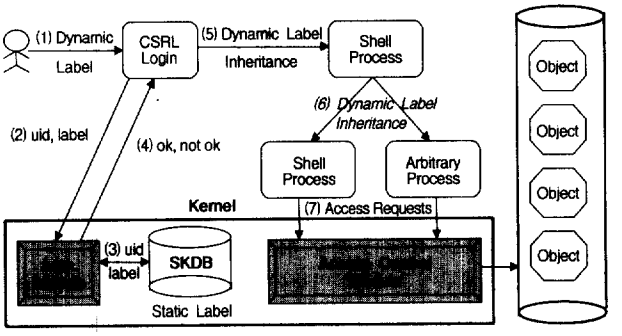
- if (s, o, a) ∈ B, ***(fp(s) = common)*** ∧ ***(fr(s) = true)*** ∧ a ∈ M[s, o].
- if (s, o, a) ∈ B, ***(fp(s) = public)*** ∧ ***(fr(s) = true)*** ∧ ***(fo(o) = shared)*** ∧ (a ∈ M[s, o]).

5. E-BLP 보안 모델 기반 CSRL

이 장에서는 본 연구팀이 개발중인 E-BLP 보안 모델을 적용한 CSRL(CSRL is SecuRe Linux) 시스템에 대해 기술한다. 현재 개발중인 CSRL시스템에서는 부서(category) 정보를 배제했다. (그림 6)은 전체적인 CSRL 접근 통제 시스템을 보여준다. 각 장의 실행결과는 Appendix를 참고하기 바란다.

5.1 주체/객체 등급화

정적 등급인 사용자 주체의 등급과 접근 객체인 파일의 등급은 6(Top Secret)에서 1(Anonymous, Shared)까지 정의했다. 각 사용자에 대한 등급은 SKDB(Security Kernel Data-Base)에 보안관리자에 의해 정의된다. 접근 객체인 파일의 등급과 실행 시 접근 주체인 프로그램들의 영역정보(Common 또는 Public)를 위해 디스크상의 inode 구조체(include/linux/ext2_fs.h)인 ext2_inode구조체의 예약된 필드(i_reserved1) 영역을 사용했으며, common영역의 프로그램들을 위해서는 정수 10을 public영역의 프로그램들을 위해서는 20을 할당했다. 그리고 객체 등급 정보를 read/write하기 위한 시스템 호출들을 추가하고, 보안 관리자를 위해 이들 시스템 호출들을 사용하는 프로그램들을 작성했다. 접근 객체인 파일의 등급은 프로세스에 의해 참조될 때 메모리 inode(include/linux/fs.h) 구조체에 추가된 필드(o_level)로 읽혀지며, 프로그램의 영역정보는 task_struct(include/linux/sched.h) 구조체에 추가된 필드(domain)로 읽혀진다. 결국 E-BLP모델에서 정의하는 주체 ***S***는 로그인 시 할당되는 사용자의 등급(s_level)과 프로세스 실행 시(exec 시스템 호출) 할당되는 영역정보로서 구성된다[Appendix A, C].



(그림 6) CSRL 접근 통제 시스템

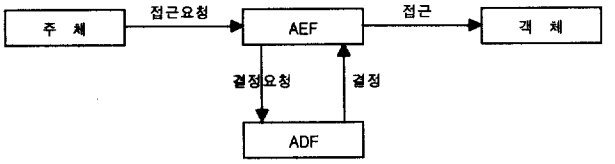
5.2 인증모듈

식별(Identification)과 인증(Authentication)은 시스템 자원을 보호하기 위한 외부의 1차적인 보호 계층이다.

CSRL 시스템의 인증절차는 다음과 같다. (그림 6)에서 (1) 등급화된 사용자는 자신의 보안 등급(또는 자신보다 낮은 보안등급)으로 시스템에 로그인을 시도한다. (2) 로그인 프로세스는 사용자가 제출한 등급과 그 사용자를 대표하는 uid (User Identifier)를 커널내의 인증모듈에 제시하고, (3,4) 인증모듈(Auth. Module)은 SKDB(Security Kernel Data Base)에서 보안 관리자에 의해 미리 정의된 uid에 해당하는 사용자의 최고 보안 등급을 검색한 후, 로그인 시 제출된 보안등급이 최고 보안등급보다 낮거나 같으면 인증허가를 로그인 프로세스에 알린다. (5) 시스템에서 인가된 사용자를 대신하고, 로그인 시 사용자의 보안등급을 상속한 Shell Process를 실행한다. 이때 사용자의 보안 등급은 리눅스에서 프로세스 디스크립터인 task_strcut 구조체의 새로 정의된 필드(s_level)에 읽혀지고, 실행되는 Shell Process의 영역정보도 새로 정의된 필드(domain)에 읽힌다. (6) 보안 등급 문제를 피하기 위해 새로 생성(fork시스템 호출)되는 프로세스들은 사용자의 보안 등급을 계속 상속받는다[Appendix B].

5.3 접근 통제 모듈

(그림 7)은 (그림 6)의 접근 통제 모듈의 접근 통제 수행을 위한 도식적인 그림이다. 주체는 객체접근에 대한 요구를 하면 AEF(Access Enforcement Facility)는 그 요구를 받아서 ADF(Access Decision Facility)에 접근결정을 요청한다. ADF는 시스템의 운영을 결정하는 정책(예 : DAC, MAC, RBAC)에 의거해 접근 허용/불허를 결정하고 그 결과를 AEF에게 알린다. AEF는 최종적으로 접근 결정을 집행한다[28]. 다시 말해 AEF는 접근 통제 메커니즘을 구현하며, ADF는 접근 통제 정책을 구현한다. 이런 정책과 메커니즘 분리의 이점은 정책 변경 시 AEF(메커니즘)에 대한 최소한의 변경으로 ADF(정책)를 수정할 수 있다는 것이다. 그러므로 다양한 정책을 수용할 수 있는 시스템의 설계 및 구현이 가능하다[29].



(그림 7) 접근 통제 기능

기존의 다양한 정책 수용을 위한 시스템들을 살펴보면 동적인 시스템 재구성을 통한 방법이 아니라 시스템을 다시 re-booting하는 형태의 정적인 방법을 택하고 있다. 본 연구팀에서 개발중인 CSRL시스템은 다양한 정책을 수용하면서 동적인 시스템의 재구성이 가능하도록 ADF구현을 모듈로서 제공한다. 리눅스 모듈이란 부팅 후에 동적으로 로드/언로드 할 수 있는 커널의 구성요소를 말한다[30, 31]. 따라서 CSRL 시스템은 각 정책에 해당하는 모듈들을 제공함으로서 필요시 동적 로드/언로드에 의해 다양한 정책을 수용할 수 있다. 여기서는 다중등급 정책을 지원하기 위한 ADF MAC모듈을 소개한다.

```
Boolean  ADF(s, o, m) {
   boolean t, e1, e2, e3,e4,e5,e6;
   if (s is in common) {
        e1 == DRC(s);
        e2 = e-ss-property(s, o, m);
        e3 = e-*-property(s, o, m);
        e4 = e-ds-property(s, o, m);
        return (e1 && e2 && e3, e4);
   }
   else if (s is in public) {
        e5 = DRC(s);
        e6 = e-ds-property check(s, o, m);
        return (e5 && e6);
   }
}
```

(그림 8) Access Decision Facility

E-BLP 모델 기반CSRL시스템의 접근 통제는 (그림 6)의 (7)과 같이 주체인 프로세스가 접근 객체에 대한 접근을 요구할 때 수행된다. AEF는 우선 접근을 요구한 주체의 영역(Common 또는 Public)을 판단하고, 접근 동작이 read 또는 write인지를 판단한 후, 각 영역에 해당하는 보안 특성 함수들(접근 결정 함수들)이 구현된 ADF 즉 MAC 모듈에게 접근 결정을 요청한다. 접근 결정의 요청을 받은 ADF는 접근 주체의 영역별로 e-ss-property 검사 또는 e-*-property 검사를 수행한다. (그림 8)은 모듈로서 제공되는 ADF MAC 모듈의 접근 결정 방식을 보여준다. 현재는 프로세스의 신뢰성 검사 루틴(DRC)을 생략했다.[Appendix D]

6. 결론 및 향후연구

다중등급 보안 운영체제에서의 기존의 접근 방법은 사용자의 등급을 실질적인 주체인 프로세스로 상속함으로써 프로세스 자체의 행위 및 신뢰성을 고려하지 않았다. 또한 실행 가능한 접근 객체들의 실행 시 보안 등급 결정 문제가 정보보호의 목적에 위배될 수 있다. 위의 문제점들을 해결하기 위해 본 논문에서는 사용자와 프로세스간 신뢰관계를 고려하고, 보안 등급 결정 문제를 해결하는 E-BLP보안 모델을 제안했고 개발중인 CSRL 시스템에 대해 소개했다.

향후 연구과제로는 모델에서 제안한 프로세스 신뢰성 검사를 위한 행위분석 및 기준마련, 다양한 정책 수용을 위한 ADF의 설계 및 구현, 감사 기록 등이 있다.

Appendix

5장에서 설명된 E-BLP 모델 기반CSRL 시스템의 실행 결과들을 보여준다.

A. 접근 객체 등급화 및 프로그램 영역지정

```
[root@themis e-blp]# ./write_object_level /etc/passwd S
The level of the /etc/passwd has been changed to TOP SECRET
[root@themis e-blp]# ./read_object_level /etc/passwd
The level of the /etc/passwd is TOP SECRET
[root@themis e-blp]# ./write_object_level /usr/bin/passwd 10
Now! The /usr/bin/passwd is in COMMON domain
[root@themis e-blp]# ./read_object_level /usr/bin/passwd
The /usr/bin/passwd is in COMMON domain
[root@themis e-blp]#
```

B. CSRL 인증 절차

```
[jmkang@seecure jmkang]$ telnet totoro.kjist.ac.kr
Trying 203.237.51.136...
Connected to totoro.kjist.ac.kr.
Escape character is '^]'.


Concurrent System Research Laboratory
Security Research Group Test-bed
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
Kernel 2.4.7 on an i686
CSRL_LOGIN: jmkang
Password:
CSRL_LEVEL: 3

Login Success!
Your Current Level is CLASSIFIED

Last CSRL_LOGIN: Fri Sep 21 01:10:38 from localhost.localdomain
[jmkang@themis jmkang]$
```

C. 실행 가능한 프로그램 실행 시 주체 정보 출력 예

다음은 프로그램 /usr/bin/passwd의 영역 정보를 COMMON으로 설정하고, 현재 3등급으로 로그인한 사용자가 /usr/bin/passwd 프로그램을 실행 시 /var/log/messages의 내용을 보여준다.

```
[root@themis /root]# tail -3 /var/log/messages
Sep 21 02:44:50 themis kernel: The information of the Subject S=<U,P> is ...
Sep 21 02:44:50 themis kernel: The User's Current level is CLASSIFIED
Sep 21 02:44:50 themis kernel: The Process is in COMMON domain
[root@themis /root]#
```

D. 접근 통제

D-1 AEF(Access Enforcement Facility) Snapshot

```
asmlinkage ssize_t sys_read(unsigned int fd, char * buf, size_t count)
{
    생략
    int Themis_ADF_result = 0;

    ret = -EBADF;
    file = fget(fd);
    if (file) {
    if (file -> f_mode & FMODE_READ) {
      /* AEF */
      if(ADF_read_file != NULL) {
        inode = file -> f_dentry -> d_inode;
        if (!(current -> s_level == 0) && !(inode -> o_level==0)) {
            /* Call ADF Module */
            Themis_ADF_result =
              ADF_read_file(current->domain,
                  current -> s_level, inode -> o_level);
        if (Themis_ADF_result == 0) {
            printk("You are not allowed to access! \n");
            return 0;
        }
      }
    }
    생략
}
```

D-2 ADF(Access Decision Facility) Module

```
/* * This Module contains the ADF(Access Decision Facility) of
   CSRL System */
#include <linux/kernel.h>
#include <linux/module.h>
#include <sys/syscall.h>
#ifdef CONFIG_MODVERSIONS
#define MODVERSIONS
#include <linux/modversions.h>
#endif
/* Decision functions declaration
   Using below names, AEFs call decision functions of this ADF. */
extern int (*ADF_read_file)(int domain, int s_level, int o_level);
extern int (*ADF_write_file)(int domain, int s_level, int o_level);
int eblp_read_file(int domain, int s_level, int o_level)
{
   printk("eblp_read_file() is called in ADF module\n");
   중간생략
   if (s_level >= o_level) {
     printk("The file read access is permitted\n");
     return 1;
   }
   else {
     printk("The file read access is denied\n");
     return 0;
   }
}
생략
```

D-3 테스트 결과

다음은 Classified등급으로 로그인 한 사용자가 Top Secret 파일인 /etc/passwd파일을 cat명령을 이용해서 읽기 접근을 시도한 경우의 커널 메시지이다.

```
[root@themis /root]# tail -11 /var/log/messages
Sep 21 05:39:41 themis kernel: --- Themis MAC module initialization ---
Sep 21 05:39:48 themis kernel: The information of the Subject S=<U,P> is ...
Sep 21 05:39:48 themis kernel: The User's Current level is CLASSIFIED
Sep 21 05:39:48 themis kernel: The Process is in COMMON domain
Sep 21 05:39:48 themis kernel: eblp_read_file() is called in ADF module
Sep 21 05:39:48 themis kernel: The Process domain is COMMON
Sep 21 05:39:48 themis kernel: The User level is CLASSIFIED
Sep 21 05:39:48 themis kernel: The Object level is TOP SECRET
Sep 21 05:39:48 themis kernel: The file read access is denied
Sep 21 05:39:48 themis kernel: You are not allowed to access!
Sep 21 05:40:41 themis kernel: ---Cleaning up the E-BLP MAC Module ---
[root@themis /root]#
```

참 고 문 헌

[1] 한국정보보호진흥원, "2001년 상반기 해킹·바이러스 분석 보고서", 2001.

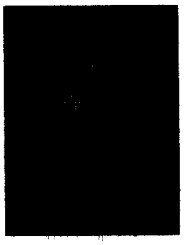
[2] NIST, "An Introduction to Computer Security : The NIST Handbook," June 20, 1994.

[3] Peter A. Loscocco, Stephen D. Smalley, Patrick A. Muckelbauer, Ruth C. Taylor, S. Jeff Turner, John F. Farrell, "The Inevitability of Failure : The Flawed Assumption of Security in Modern Computing Environments," Technical report, United Stated National Security Agency (NSA), 1995.

[4] Bill Neugent, "Where We Stand in Multilevel Security (MLS) : Requirements, Approaches, Issues, and Lessons Learned," Computer Security Applications Conference, 1994.

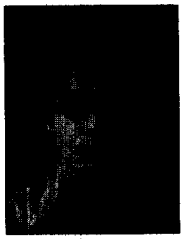
[5] Len Lapadula, "Secure Computer Systems : Mathematical

Foundations," MITRE Technical Report, Vol. I, 1996.
[6] Len Lapadula, "Secure Computer Systems : Mathematical Foundations," MITRE Technical Report, Vol.II, 1996.
[7] Carl E. Landwehr, "Formal Models for Computer Security," ACM Computing Surveys, Vol.13, No.3, 1981.
[8] Carl E. Landwehr, Constance L. Heitmeyer, and John McLean, "A Security Model for Military Message Systems," ACM Trans., Aug. 1984.
[9] Frank L. Mayer, "An Interpretation of a Refined Bell-La Padula Model For the TMach Kernel," Aerospace Computer Security Applications Conference, 1988.
[10] John McLean, "The Specification and Modeling of Computer Security," Computer, Volume : 23, Issue : 1, 1990.
[11] DOD 5200.28-STD, "Trusted Computer System Evaluation Criteria," December 1985.
[12] 홍기용 외, "안전한 운영체제를 위한 MAC메커니즘의 설계 및 구현", 한국정보과학회 가을학술발표논문집, Vol.17, No.2, 1990.
[13] Peter A. Loscocco, Stephen D. Smalley, Patrick A. Muckelbauer, Ruth C. Taylor, S. Jeff Turner, John F. Farrell, "Integrating Flexible Support for Security Policies into the Linux Operating System," Technical report, United Stated National Security Agency (NSA), 2000.
[14] Warwick Ford, "Computer Communications Security," Prentice Hall, 1994.
[15] Edward G. Amoroso, "Fundamentals Of Computer Security Technology," Prentice Hall, 1994.
[16] Dieter Gollman, "Computer Security," Jone Wiley & Sons, 1999.
[17] Michael V. Joyce, "Access Control and Applications on Trusted Systems," Computer Security Applications Conference, 1992.
[18] R. S. Sandhu, and P. Samarati., "Access Control : Principles and Practices," IEEE Communications, 1994.
[19] R. S. Sandhu, and P. Samarati, "Authentication, Access Control, and Audit," ACM Computer Survey, 28, 1, 1996.
[20] David Ferraiolo, and Richard Kuhn, "Role-Based Access Control," Proceedings of 15th National Computer Security Conference, 1992.
[21] D. Ferraiolo, J. Cugini, and K. Richard, "Role-Based Access Control (RBAC) : Features and Motivations," In Proc. of the Annual Computer Security Applications Conference, 1995.
[22] R. Sandhu, E. Coyne, H. Feinstein, "Role-Based Access Control Models," Computer 29(2), 1996.
[23] Raymond M. Wong, "A Comparison of Secure UNIX Operating Systems," Computer Security Applications Conference, 1990.
[24] http : //www.rsbac.de.
[25] http : //www.nsa.gov/selinux/.
[26] 강정민, 신 욱, 박춘구, 이동익, "프로세스 신뢰도에 기반한 확장된 BLP 보안 모델과 아키텍처 설계", 한국정보과학회 춘계학술대회논문집, 2001.
[27] Daniel F. Stern, and Glenn S. Benson, "Redrawing the Security Perimeter of a Trusted System," Computer Security Foundations Workshop Ⅶ, 1994.
[28] ITU-T, "Security Frameworks For Open Systems : Access Control Framework," ITU-T Recommendation X.812, 1996.
[29] Grenier, G.-L. ; Holt, R.C. ; Funkenhauser, M., "Policy VS. Mechanism in the Secure TUNIS Operating System," Proceedings. of IEEE Symposium on Security and Privacy, 1989.
[30] A. Rubini and J. Corbet, "Linux Device Drivers," O'REILLY, 2001.
[31] D. P. Bovet and M. Cesati, "Understanding Linux Kernel," O'REILLY, 2001.



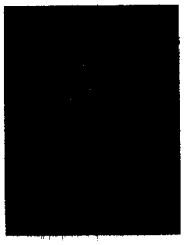
강 정 민

e-mail : jmkang@kjist.ac.kr
2000년 인제대학교 전산학과(학사)
2001년~현재 광주과학기술원 정보통신공학과 (석사과정)
관심분야 : 정보보호, 이동 컴퓨팅, 분산 시스템



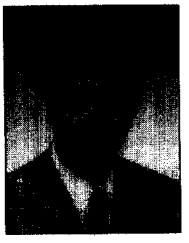
신 욱

e-mail : sunihill@kjist.ac.kr
1998년 동국대학교 컴퓨터공학과(학사)
2000년 광주과학기술원 정보통신공학과(석사)
2000년~현재 광주과학기술원 정보통신공학과 (박사과정)
관심분야 : 운영체제 보안, 안전성 검증 기법 등



박 춘 구

e-mail : sunihill@kjist.ac.kr
2000년 인하대학교 전자계산공학과(학사)
2001년~현재 광주과학기술원 정보통신공학과 (석사과정)
관심분야 : 정보보호, Bioinformatrics 등



이 동 익

e-mail : dilee@kjist.ac.kr
1985년 영남대학교 전기공학과(학사)
1989년 오사카대학 전자공학과(석사)
1993년~1994년 일리노이 대학 컴퓨터공학과 방문연구원
1990년~1995년 오사카 대학 전자공학과 문부교관
1995년~현재 광주과학기술원 정보통신공학과 부교수
관심분야 : 병행시스템(Concurrent Systems) 해석 및 설계, Petri Nets 이론, 이동 에이전트 시스템, 정보보호, 비동기 회로 설계 및 CAD 등