

교과목 : 정보보호

7. Access Control

2023학년도 2학기
Suk-Hwan Lee

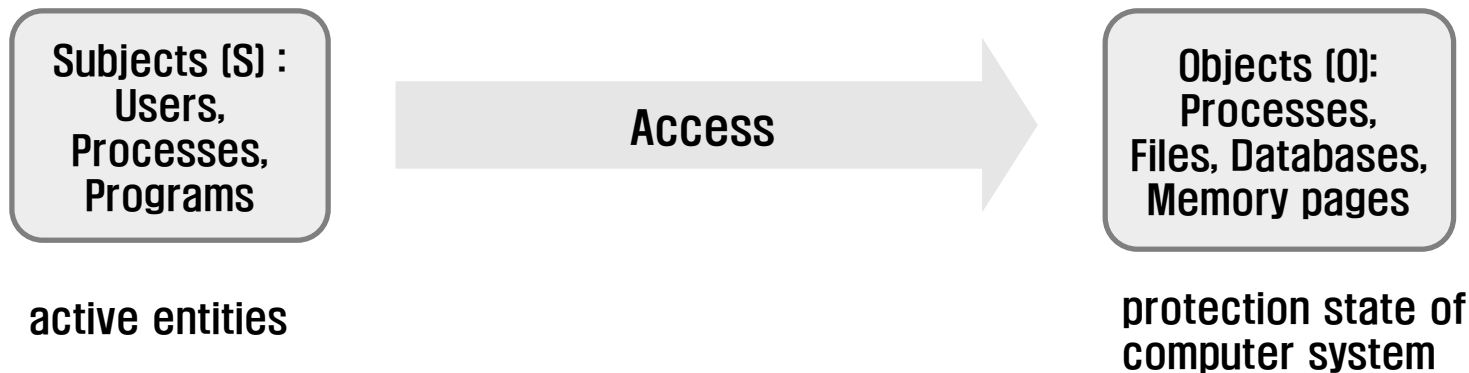


- **참고자료**

- ✓ 황성운 저, 정보 보안 원리 및 실습, 2장 접근제어
- ✓ [주교재] Mark Stamp, Information Security: Principles and Practice, 2nd edition

- 접근제어 (Access Control) 란

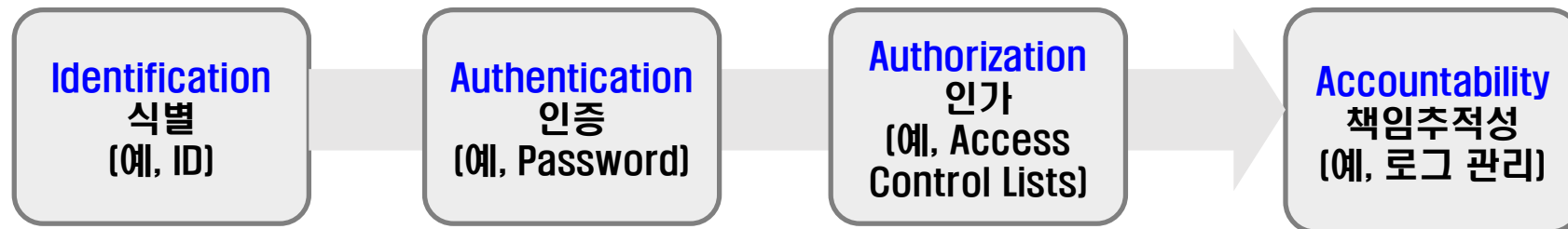
- ✓ 자원이 어떻게 접근되는지를 제어하여 인가되지 않는 수정이나 노출로부터 보호하는 것을 말한다.
- ✓ 접근은 **주체**와 **객체** 사이의 흐름
 - 주체(Subjects)는 사용자, 프로그램 혹은 프로세스와 같은 활성화된 요소
 - 객체(Objects)는 컴퓨터, 데이터베이스, 파일, 컴퓨터 프로그램 등과 같은 수동적인 요소
 - 주체는 작업을 수행하기 위해 객체 혹은 객체 안의 데이터에 대한 접근을 요청한다.
 - 예를 들어, 프로그램이 파일에 접근하면, 프로그램은 주체가 되고 파일은 객체가 된다.



- 접근제어 (Access Control) 필요성

- ✓ 시스템과 네트워크 등 자원에 시도되는 허가되지 않은 접근에 대응하기 위한 첫 번째 방어 수단 중 하나로,
- ✓ 조직이나 시스템에서 **자원의 가용성, 무결성, 기밀성을 보호**할 수 있게 도와주는 역할을 수행한다.

- 접근제어 실행 단계



- 접근제어 실행 단계

- ✓ 식별(Identification)

- 주체가 인증 서비스에 스스로를 확인시키기 위하여 자신의 신원 정보를 제공하는 활동
- 신원 정보는 물리적 시설 또는 컴퓨터 정보 시스템에 접근을 가능하게 하는 물리적인 객체, 지식 또는 사람이 가지고 있는 특성을 말함
- [예:사용자명, 계정 번호, 메모리 카드, 홍채/지문 등과 같은 생체 인식 정보].

- ✓ 인증(Authentication)

- 주체가 자신의 신원을 증명하기 위해서 행하는 검증 활동을 말함
- 대표적 예; 사용자가 아이디와 패스워드를 가지고 인증
- 사용자의 신원 인증(PIN, 토큰, 스마트카드, 생체인증(지문, 정맥), 움직임, 음성, 터치, 서명 등), 장치 인증(IP나 MAC 주소 기반), 응용 프로그램 인증(TCP/UDP 포트 번호)

- 접근제어 실행 단계

- ✓ **인가(Authorization, 권한 부여)**

- 인증된 주체가 자원에 대한 접근 요청을 판단하여 허용하는 것
- 주체가 가지고 있는 보안 수준 보안 수준 및 알 필요성 등 **보안 정책**을 참조하며 **접근 제어 목록, 보안 등급** 등이 사용

- ✓ **책임추적성(Accountability)**

- 시스템에 접근한 주체가 시스템에 어떤 행위를 하고 있는지를 기록함으로써, 문제 발생 시 원인 및 책임 소재를 파악하기 위함
- 책임추적을 위해 감사(Audit), 디지털 포렌식 등의 방법이 사용

- 접근 제어 원칙

- ✓ 직무 분리(Separation of Duties)

- 중요한 정보나 민감한 정보를 다루는 기능을 한 사람이나 그룹이 다하지 못하게 (단계별로) 분리함으로써 접근 권한을 남용하는 것을 막거나 최소화하는 것을 목표로 한다.
- 예로, 시스템 관리자가 자신이 관리하는 서버에서 발생하는 활동을 스스로 모니터링 하는 대신에, 객관적인 제3자, 예를 들어 보안팀 구성원 등이 모니터링 하도록 하는 것이다.

- ✓ 최소 권한(Least Privilege, 최소 특권)

- 어떤 사람에게도 주어진 업무를 수행하는 데 필요 이상의 권한을 부여해서는 안 된다는 것을 의미한다. 이 원칙하에서는 최소한의 권한이 주어져야 하며 기본(Default) 권한이 주어져서는 안 된다.
- 예를 들어, 구매 담당 직원은 구매 업무 이외의 파일에 접근해서는 안 된다. 운영체제에서 사용하는 관리자 계정(리눅스 시스템에서의 root, 윈도우 시스템에서의 administrator는 이 원칙이 적용되지 않는 예이다.

- 접근 제어 원칙

- ✓ 알 필요성(Need to Know)

- 주체가 일을 수행하기 위해 객체에 접근할 필요가 없다면, 그 주체는 객체에 접근 권한을 가져서는 안 된다. 예를 들어, 주체가 객체에 정보를 첨부하되, 객체 내의 정보를 수정할 필요가 없다면, 그 주체는 쓰기 권한을 받아서는 안 되고 첨부 권한만을 받아야 한다.

- ✓ 고장시 안전 초기화(Fail-Safe Default)

- 어떤 주체가 객체에 대해서 명시적으로 접근 권한을 부여 받지 않았다면, 그 주체는 기본적으로 객체에 대한 접근이 금지되어야 한다. 이 원칙은 또한 주체가 주어진 일을 완료하지 못하고 실패하더라도, 그 주체가 행한 변화를 초기 상태로 되돌림으로써 시스템을 안전한 상태로 유지해야 한다는 것을 의미한다.
 - 예를 들어, 라우터는 패킷이 들어오면 액세스 리스트(Access List)의 처음부터 맨 마지막까지 탐색하는데 일치하는 것이 없으면 라우터는 접근이 허용되지 않는 것으로 판단하여 해당 패킷을 버리게 된다.

- 접근 제어 원칙

- ✓ 완전한 중재(Complete Mediation)

- 주체의 객체에 대한 접근은 처음뿐만 아니라 항상 검사되어야 한다. 이 원칙은 많은 시스템에서 서비스 구현을 용이하게 하기 위해 위배되는 경향이 있다. 예를 들어, 프로세스가 파일에 대한 접근을 요청하는 경우, 유닉스 운영체제에서는 어떤 프로세스가 파일에 접근할 때 처음에는 커널에서 접근을 검사하나 한번 접근이 허용된 이후에는 파일 소유 정보가 변경되어 해당 프로세스에게 접근이 허용하지 않을지라도 커널은 여전히 접근을 허용한다.

- 고장시 안전 초기화 (Fail-Safe Default)

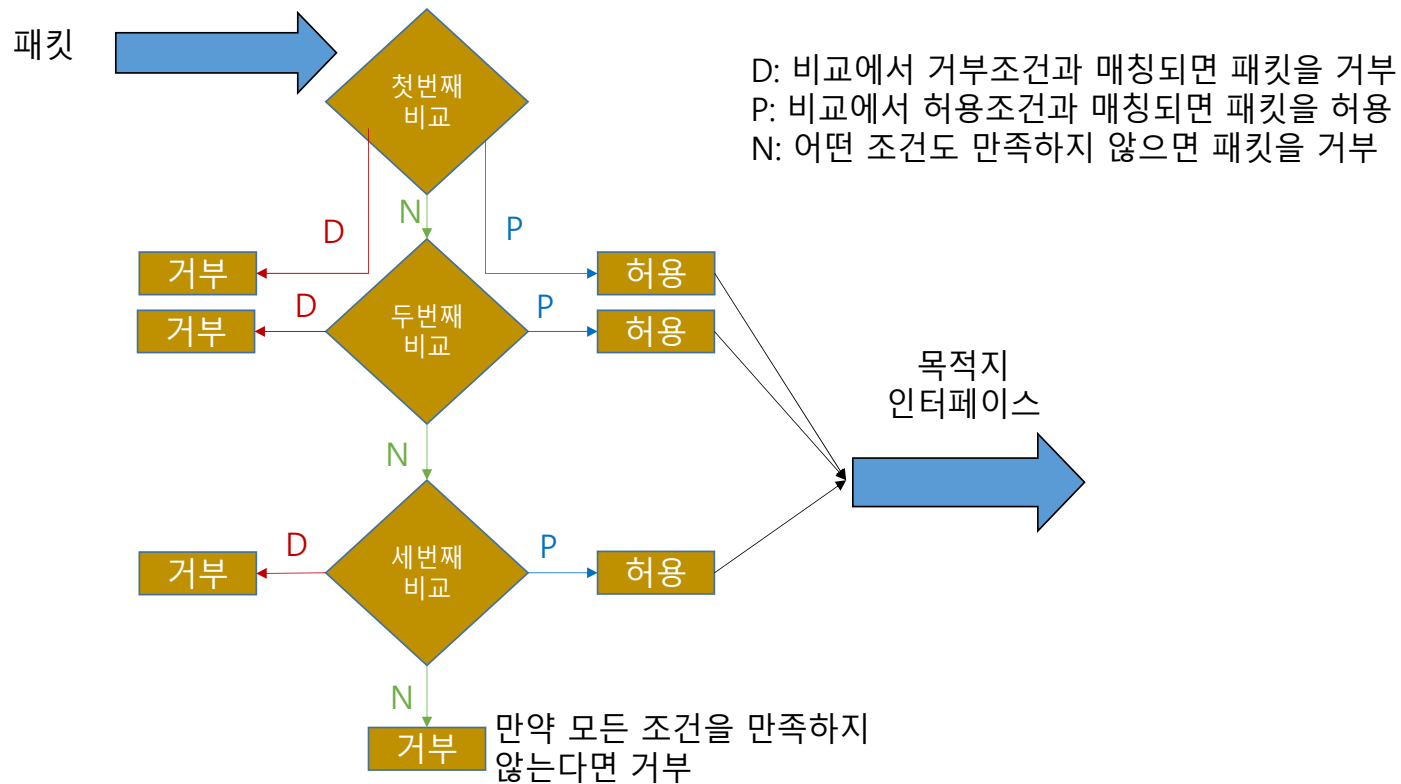


그림. 라우터에서의 액세스 리스트

Access Control Mechanism

- 접근 제어 행렬(Access Control Matrix)

- ✓ 주체 S가 객체 O에 허용된 접근은 S와 O에 해당하는 행과 열이 교차하는 위치에 저장된다.
- ✓ 실제 시스템 환경에서는 주체와 객체의 개수가 매우 크기 때문에, 주체가 객체를 참조할 때마다 이렇게 방대한 행렬을 참조하는 것은 부담이 된다. 따라서, 아래와 같이 행과 열로 분리해서 사용하는 경우가 많다.

- 접근 가능 행렬(Capability List, 권한 목록, 자격 목록)

- ✓ 접근 제어 행렬을 행 단위로 분해한 것으로, 한 주체가 접근 가능한 객체와 권한을 명시하는 목록을 말한다. 커버로스 및 안드로이드를 포함한 분산 시스템 환경에서 많이 사용되는 기법이다.

- 접근 제어 목록(ACL: Access Control List)

- ✓ 접근 제어 행렬을 열 단위로 분해한 것으로, 한 객체에 대해 접근 가능한 주체와 권한을 명시하는 목록을 말한다. 운영체제, 응용 프로그램, 라우터, 방화벽 등에서 일반적으로 많이 사용되는 기법이다.

- 접근 제어 행렬(Access Control Matrix)

접근 제어 목록

↓

	파일 A	파일 B	파일 C
영희	rx	r	r
철수	rx	rx	r
관리자	rwX	rwX	r

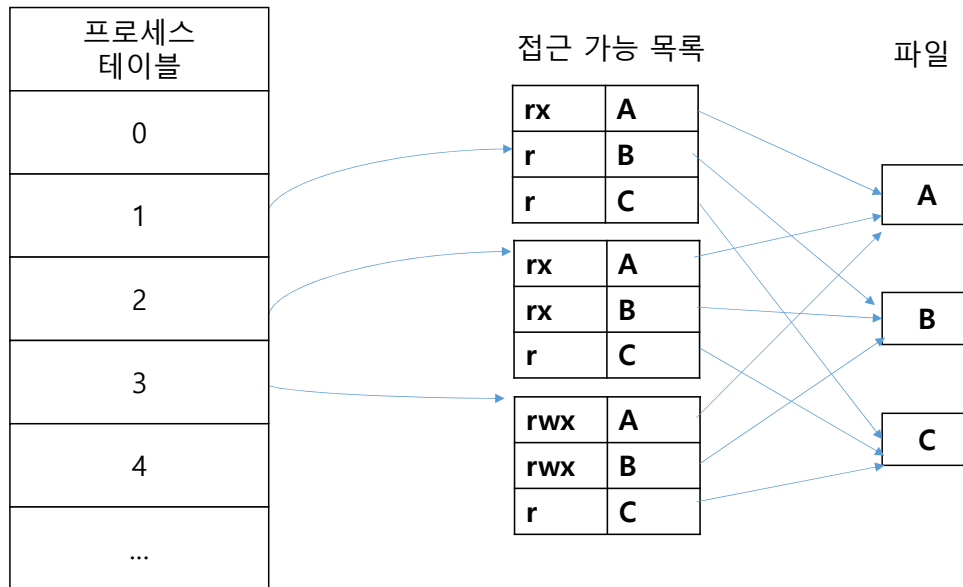
접근 가능 목록 →

- 접근 가능 목록

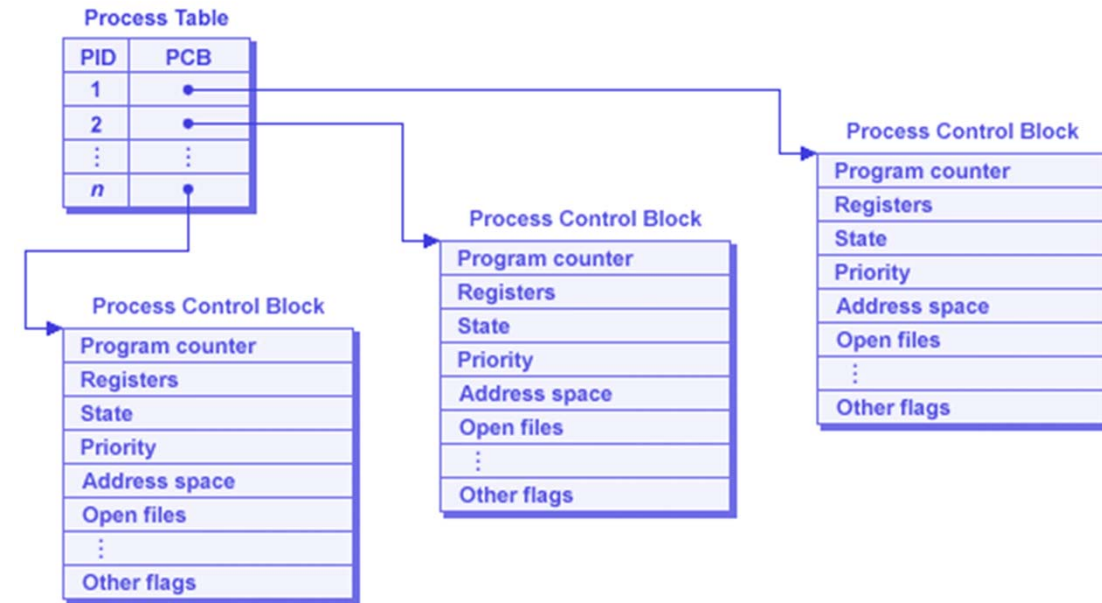
영희: [파일 A: rx]; [파일 B: r]; [파일 C: r];
철수: [파일 A: rx]; [파일 B: rx]; [파일 C: r];
관리자: [파일 A: rwX]; [파일 B: rwX]; [파일 C: r];

- 접근 제어 목록

파일 A: [영희: rx]; [철수: rx]; [관리자: rwX];
파일 B: [영희: r]; [철수: rx]; [관리자: rwX];
파일 C: [영희: r]; [철수: r]; [관리자: r];



Linux Process Table



A Process identification number (PID) is assigned to each process


A Process control block (PCB) is used to track the process' s execution status.

[참고] <https://www.geeksforgeeks.org/process-table-and-process-control-block-pcb/>

Access Control Model

- **Access control model** : a framework that dictates how subjects access objects
 - Use access control technologies and security mechanisms to enforce the rules and objectives of the model
 - Three main types of access control models:

- ✓ **Discretionary Access Control (DAC)**, 임의적 접근 제어
- ✓ **Mandatory Access Control (MAC)**, 강제적 접근 제어
- ✓ **Role-Based Access Control (RBAC)**, 역할 기반 접근 제어

- 
- How to choose a access control model?
 - ✓ Business and security goals, Culture of the company, Habits of conducting business,
 - ✓ Some companies use one model exclusively, others combine them

➤ Discretionary Access Control (DAC) : 임의적 접근 제어 (ID 기반 접근 제어)

▪ File and Data ownership

- ✓ 시스템 상 모든 자원 객체는 소유자가 있음. 대부분 각 객체의 처음 소유자는 객체 생성한 주체이며, 객체의 접근 정책은 소유자에 의하여 정해짐

▪ Access rights and permissions

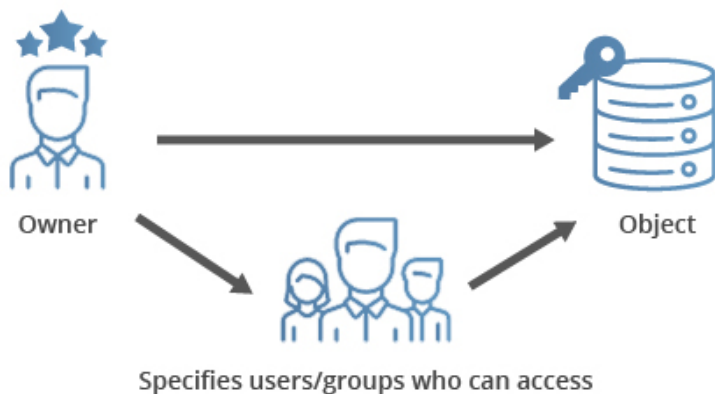
- ✓ 시스템에서 자원의 소유자가 자신의 자원에 접근할 수 있는 주체(specific users or groups of users)를 access permissions 지정 가능
- ✓ 예: 데이터 소유자는 자신의 파일에 Bob(사용자 식별자)과 회계 부서(그룹 식별자)가 접근하도록 지정할 수 있다.
- ✓ Access permissions for each piece of data are stored in an **access-control list (ACL)**

▪ 구현이 쉽고 권한 변경이 유연하나, 주체별로 객체에 대한 접근 권한을 부여해야 하므로 불편하며 주체와 객체의 수가 증가함에 따라 규칙의 수도 기하급수적으로 증가하는 문제 발생

- ✓ 예: 오늘날 대부분의 운영체제(윈도우, 리눅스, 맥, 유닉스 시스템 등)에서 채용
- ✓ 유닉스 시스템에서의 사용자/그룹 계정, 읽기-쓰기-실행 권한은 대표적인 예

➤ Discretionary Access Control (DAC) : 임의적 접근 제어 (ID 기반 접근 제어)

Discretionary Access Control (DAC)



Gaining access in the DAC model works like this:

- User 1 creates a file and becomes its owner or obtains access rights to an existing file.
- User 2 requests access to this file.
- User 1 grants access at their own discretion. However, user 1 can't grant access rights that exceed their own. For example, if user 1 can only read a document, they can't allow user 2 to edit it.
- If there's no contradiction between the ACL created by an administrator and the decision made by user 1, access is granted.

[참고] <https://www.ekransystem.com/en/blog/mac-vs-dac>

➤ Mandatory Access Control (MAC) : 강제적 접근 제어

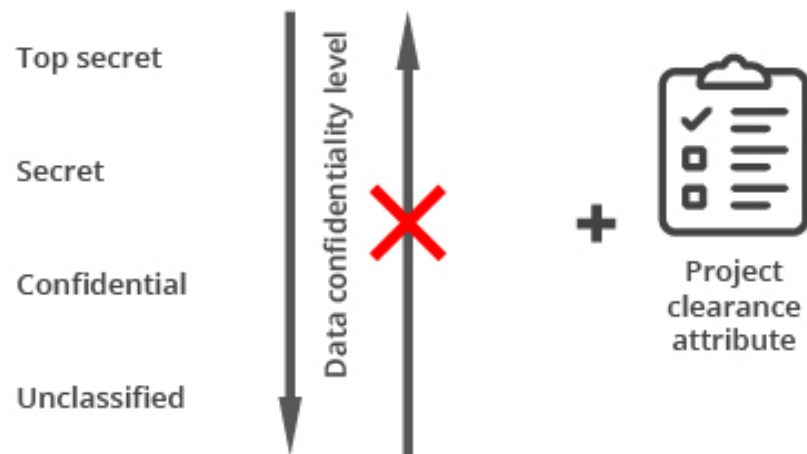
- 시스템 전체적으로 적용되는 메커니즘(보안 정책 및 관련 규칙)에 따라 **관리자에 의해서 접근 제어가 결정됨**
 - ✓ 개인이나 소유자 등 개별 주체에 의해 결정되지 않음
- 시스템의 모든 주체와 객체가 **보안 레이블**을 반드시 가져야 함
 - ✓ 주체는 **비밀 취급 허가 수준(Clearance Level)**과 **알 필요성 수준**, 객체는 **보안 분류 수준(Classification Level)**과 **범주(Classification, Compartment)**를 가짐
 - ✓ 비밀 취급 허가 수준은 주체가 다룰 수 있는 객체의 민감성을, 보안 분류 수준은 객체가 갖고 있는 민감성을 나타냄 (Top Secret > Secret > Confidential > Unclassified)
 - ✓ 범주는 정보를 분류하는 기준으로 부서, 프로젝트, 관리 수준 등이 될 수 있음

➤ Mandatory Access Control (MAC) : 강제적 접근 제어

- 보안 레이블에 할당된 주체의 비밀 취급 허가 수준과 알 필요성 수준을 객체의 보안 분류 수준 및 범주와 비교하여 접근 허용 여부
 - ✓ 만약 주체의 비밀 취급 허가 수준이 객체의 분류 수준보다 높거나 동등하고, 주체의 알 필요성 수준이 객체의 범주를 포함되는 경우에만 접근이 허용된다.
 - ✓ [주의] 임의적 접근 제어 모델에서는 시스템이 자원의 접근 제어 목록과 접근하려는 주체의 식별자를 비교하여 결정한다.
- 기밀 문서가 엄격히 다루어져야 하는 군이나 정부 정보기관 등 중앙집중형 보안 관리에 적합
 - ✓ 예: SELinux는 기존의 임의적 접근 모델을 지원하는 리눅스 시스템에 부가적으로 강제적 접근 제어 모델을 부가 지원

➤ Mandatory Access Control (MAC) : 강제적 접근 제어

Mandatory access control

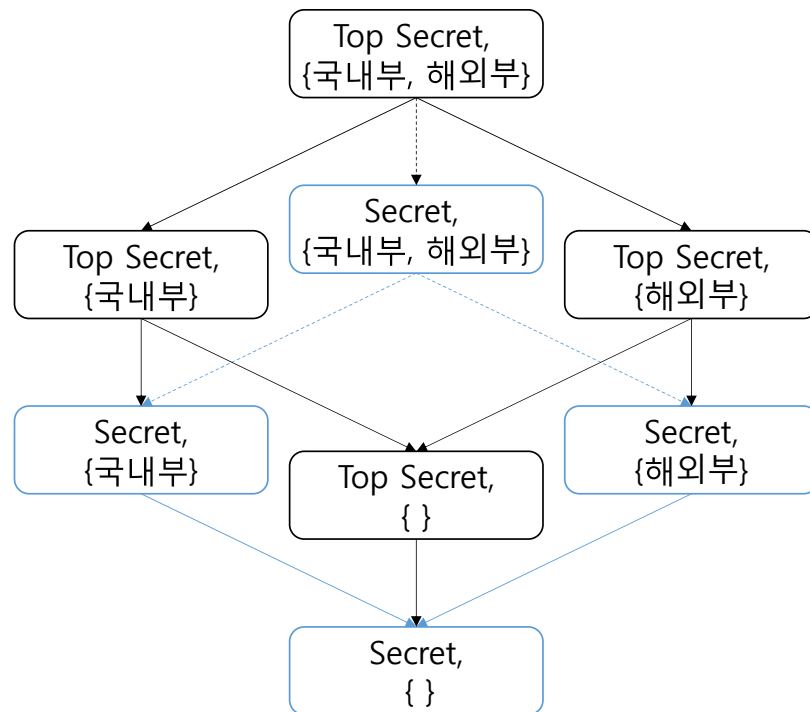


With MAC, the process of gaining access looks like this:

- The administrator configures access policies and defines security attributes: confidentiality levels, clearances for accessing different projects and types of resources.
- The administrator assigns each subject (user or resource that accesses data) and object (file, database, port, etc.) a set of attributes.
- When a subject attempts to access an object, the operating system examines the subject's security attributes and decides whether access can be granted.

[참고] <https://www.ekransystem.com/en/blog/mac-vs-dac>

➤ Mandatory Access Control (MAC) 예시



- <Top Secret, {국내부, 해외부}> 레벨에 있는 주체는 레티스에 있는 모든 객체에 접근할 수 있다.
- <Secret, {국내부}> 레벨에 있는 주체는 <Top Secret, {국내부}> 레벨에 있는 객체에 접근할 수 없다.
- 왜냐하면 부여받은 비밀 취급 허가 수준이 객체에 할당된 분류 수준과 동등하거나 높지 않기 때문이다.
- Top Secret 비밀 취급 허가 수준을 갖고 있다고 해서 이것이 조직 내의 모든 Top Secret 정보에 대한 접근 권한을 갖고 있다는 것을 의미하지는 않는다.
- <Top Secret, {국내부}> 레벨에 있는 주체가 <TopSecret, {국내부, 해외부}> 레벨에 있는 객체에 접근할 수 있는 것이 아니다. 알 필요성을 만족시키지 못하기 때문이다.

➤ MAC & DAC

Pros and cons of MAC



Pros

- **High level of data protection** — An administrator defines access to objects, and users can't edit that access.
- **Granular** — An administrator sets user access rights and object access parameters manually.
- **Immune to Trojan Horse attacks** — Users can't declassify data or share access to classified data.



Cons

- **Maintainability** — Manual configuration of security levels and clearances requires constant attention from administrators.
- **Scalability** — MAC doesn't scale automatically.
- **Not user-friendly** — Users have to request access to each new piece of data; they can't configure access parameters for their own data.

Pros and cons of DAC



Pros

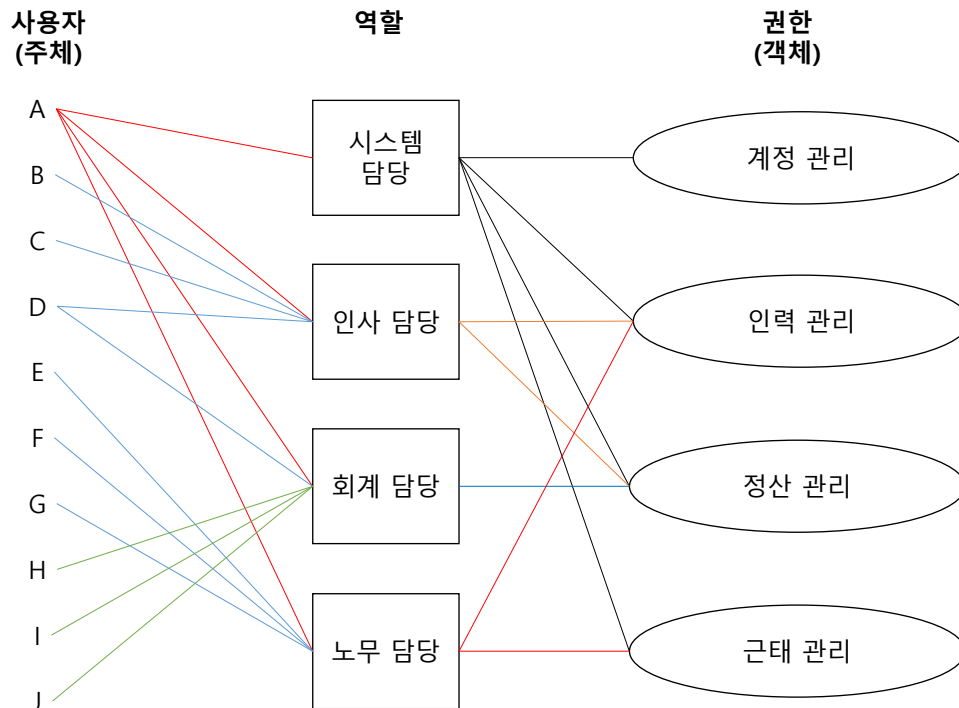
- **User-friendly** — Users can manage their data and quickly access data of other users.
- **Flexible** — Users can configure data access parameters without administrators.
- **Easy to maintain** — Adding new objects and users doesn't take much time for the administrator.
- **Granular** — Users can configure access parameters for each piece of data.



Cons

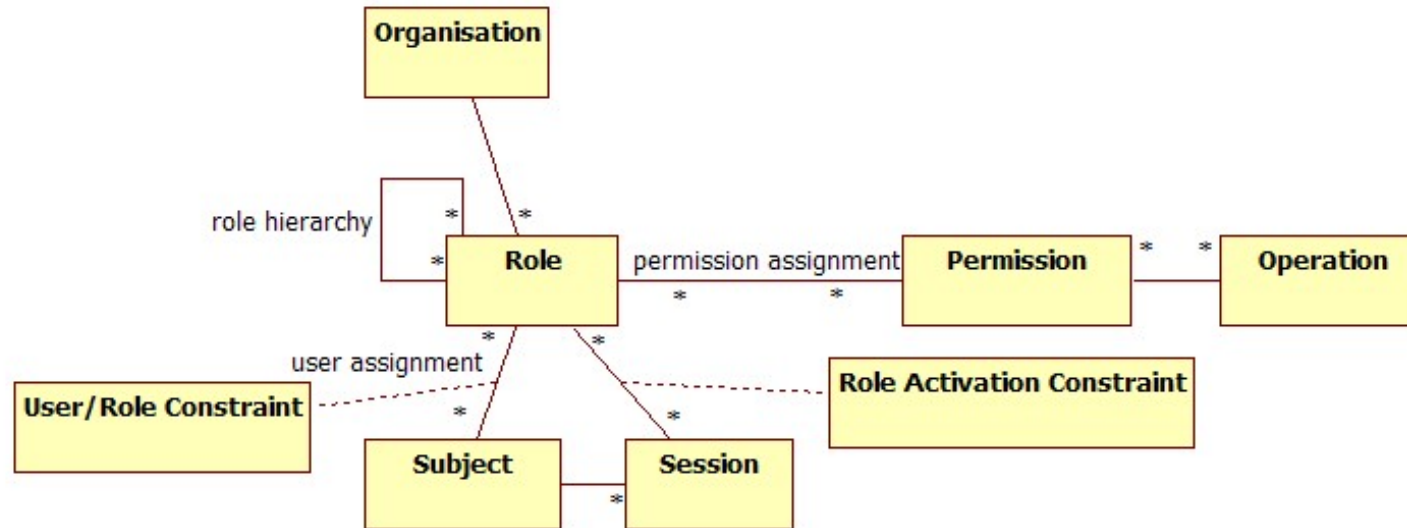
- **Low level of data protection** — DAC can't ensure reliable security because users can share their data however they like.
- **Obscure** — There's no centralized access management, so in order to find out access parameters, you have to check each ACL.

➤ Role-Based Access Control (RBAC), 역할 기반 접근 제어



- 사용자들을 직무에 따라 역할로 그룹핑하고 이 역할에 접근 권한을 부여
 - ✓ 사용자들에게 직접적으로 접근 권한을 할당하지 않음
 - ✓ 역할은 조직에서 일어나는 활동을 자연스럽게 모델링하며, 인사이드가 잦은 조직에 효과적으로 대응할 수 있어 비즈니스 환경(예: 은행, 보험, 병원 등)에서 많이 사용
- SELinux(Security-Enhanced Linux)은 기존의 임의적 접근 제어 모델, 강제적 접근 제어 모델 외에도 역할 기반 접근 제어 모델도 지원

SELinux (보안 강화 리눅스) 미국 국방부 스타일의 강제 접근 제어(MAC)를 포함한 접근 제어 보안 정책을 지원하는 매커니즘을 제공하는 리눅스 커널 보안 모듈임. <https://github.com/SELinuxProject/selinux>

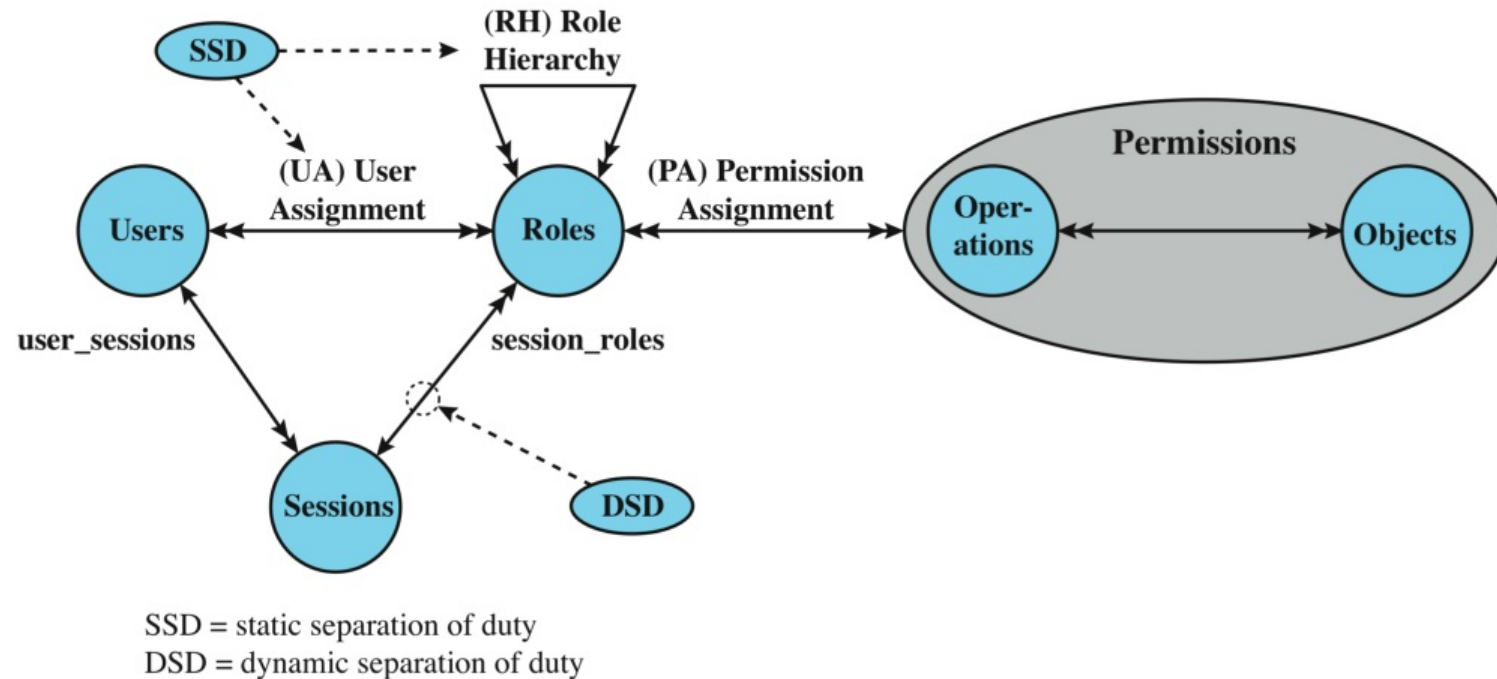


- Three primary rules are defined for RBAC:
 - ✓ **Role assignment:** A subject (S) can exercise a permission (P) only if the subject has selected or been assigned a role (R).
 - ✓ **Role authorization:** A subject's active role (R) must be authorized for the subject (S). With rule 1 above, this rule ensures that users can take on only roles for which they are authorized.
 - ✓ **Permission authorization:** A subject (S) can exercise a permission (P) only if the permission is authorized for the subject's active role. With rules 1 and 2, this rule ensures that users can exercise only permissions (P) for which they are authorized.

➤ NIST RBAC Model

- The NIST/ANSI/INCITS RBAC standard (2004) recognizes three levels of RBAC:

1. **core RBAC**
2. **hierarchical RBAC**, which adds support for inheritance between roles
3. **constrained RBAC**, which adds separation of duties



➤ Access Control Model 선택

- 보안 정책 개발을 수립 후 이를 구현할 방안을 결정할 때 고려할 접근 모델 선정 가이드라인
 - ✓ 임의적 접근 제어 모델은 데이터 소유자가 다른 사용자들에게 자신의 자원에 접근할 수 있도록 허락하므로, 조직은 어떤 결과를 가져올 것인지를 충분히 인지할 때 이 모델을 선택
 - ✓ 구축 환경이 더욱 높은 수준의 보안을 요구하거나 오직 관리자만이 자원에 대한 접근 허가를 주어야 할 경우에는 강제적 접근 제어 모델이 적합
 - ✓ 만약 조직이 높은 이직률을 보이거나 중앙 집중식 접근 모델 기법이 필요하다면 역할 기반 접근 제어 모델이 적합
- 조직이나 업무, 시스템의 특성에 따라 이들을 적절히 조합해서 사용 가능함

Security Policy

- **보안 정책**
 - ✓ 보안 시스템이 만족해야 할 추상적인 목표를 기술
 - ✓ 기밀성, 무결성, 가용성 등
- **보안 모델**
 - ✓ 목표를 수행하기 위해 필수적으로 해야 될 것을 수학적 공식 또는 아키텍처 형태로 표현
 - ✓ 종류
 - Bell-La Padula (BLP) 모델
 - BIBA 모델

➤ Bell-La Padula (BLP) 모델

- Bell-LaPadula Model (BLP)

- ✓ A state machine model (계산의 수학적 모델) used for enforcing access control in government and military applications. (Bell과 LaPadula에 의해 제안된 접근 통제 모델)

- 기밀성을 보장하기 위해 다음과 같은 주요 규칙 사용

1) 단순 보안 규칙(Simple Security Rule)

- ✓ **No Read Up Rule (상향 읽기 금지)** : 주체가 객체를 읽기 위해서는 주체의 비밀 취급 허가 수준이 객체의 보안 분류 수준보다 높거나 같아야 한다
 - 예: 홍길동이 비밀 취급 허가 수준이 Secret일 때 홍길동은 Top Secret로 분류된 데이터를 읽을 수 없다.
 - 그렇지 않으면, 데이터가 높은 보안 수준으로부터 낮은 보안 수준으로 흘러서 기밀성을 훼손하게 된다.

2) Star 보안 규칙(*-Security Rule)

- ✓ **No Write Down Rule (하향 쓰기 금지)** : 주체가 객체에 쓰기 위해서는 주체의 비밀 취급 허가 수준이 객체의 보안 분류 수준보다 낮거나 같아야 한다
 - 예: 홍길동이 비밀 취급 허가 수준이 Top Secret일 때 홍길동은 자신이 아는 내용을 Secret로 분류된 파일에 쓰지 못하게 되어 있다.
 - 이것은 Top Secret 보안 수준을 가진 사람이 우연히 Secret 등급인 파일에 기록함으로써 Top Secret인 파일의 내용이 누설되는 것을 막기 위함이다.

3) Strong Star 보안 규칙(Strong *-Security Rule)

- ✓ **읽기와 쓰기 능력**을 가지고 있는 주체는 이러한 기능을 하위 혹은 상위가 아닌 **동일한 수준에서만 수행**할 수 있다고 규정한다.
- ✓ 객체에 대해서 읽고 쓰기를 할 수 있는 주체는 비밀 취급 허가 수준과 데이터(객체) 보안 분류 수준이 동일해야 한다.

➤ Bell-La Padula (BLP) 모델

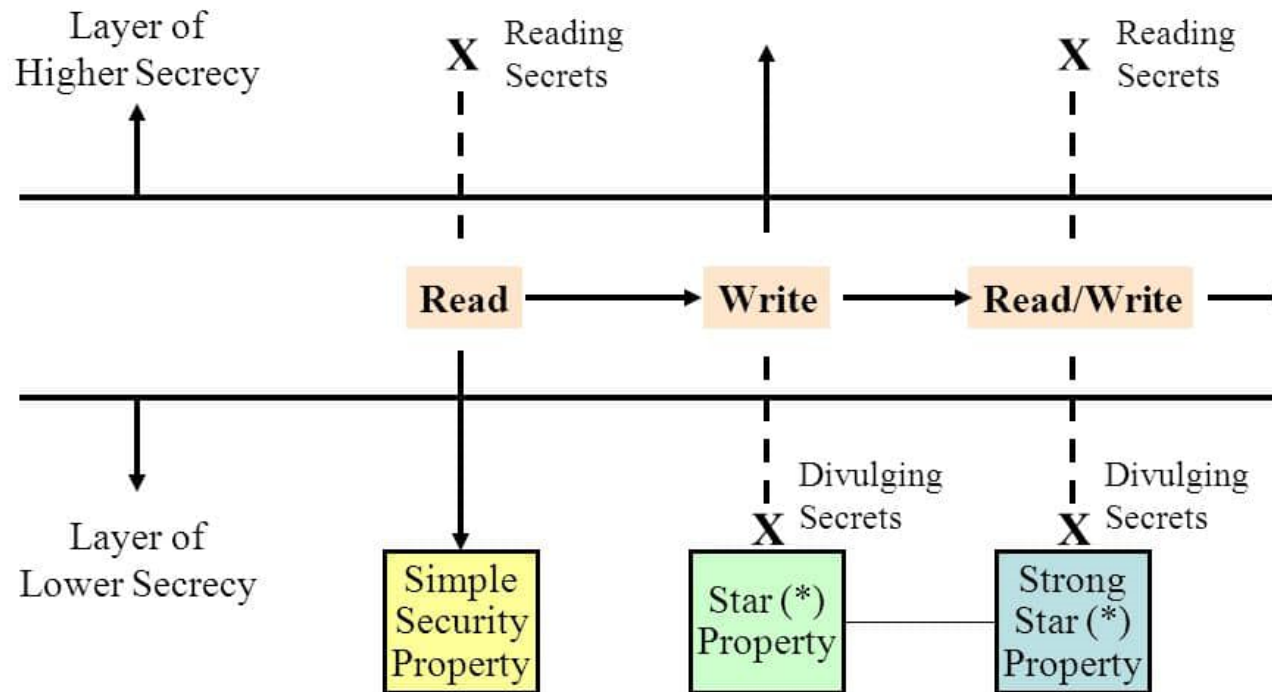


그림. BLP 모델의 세 가지 주요 규칙

➤ BIBA 모델

- Biba model or Biba integrity model
 - ✓ A formal **state transition system** of computer security policy that describes a set of access control rules designed to **ensure data integrity**.
 - ✓ → 주체 및 객체에 무결성 등급 부여가 가능한 무결성 제약 조건 보장 모델
- 무결성(부적절한 변조 방지)을 보장하기 위해 다음과 같은 주요 규칙 사용
 - ✓ BLP 모델의 규칙을 뒤집음으로써 얻을 수 있음
 - ✓ 즉, No Read Up; No Write Down 규칙을 No Read Down, No Write Up으로 바꿈

➤ BIBA 모델

1) 단순 무결성 규칙(Simple Integrity Rule)

- ✓ **No Read Down** : 주체는 자신보다 낮은 무결성 수준의 데이터를 읽을 수 없다.
 - 그렇지 않으면 주체가 낮은 무결성 수준의 손상된 데이터를 읽어 들여 높은 무결성 수준의 데이터를 훼손할 수도 있다.
 - 예: 낮은 수준의 사용자가 특권 프로세스의 실행 경로 상에 Trojan을 도입했을 때, 특권 프로세스는 시스템에 해를 입히지 않게 된다. 왜냐하면, 이 특권 프로세스는 저수준 실행물(즉, Trojan)에 접근할 수 없기 때문이다.

2) Star 무결성 규칙(*- Integrity Rule)

- ✓ **No Write Up** : 주체는 자신보다 높은 무결성 수준에 있는 객체를 쓸 수 없다
- ✓ 높은 무결성 수준으로 쓰기 권한을 행사할 수 없다.

➤ BIBA 모델

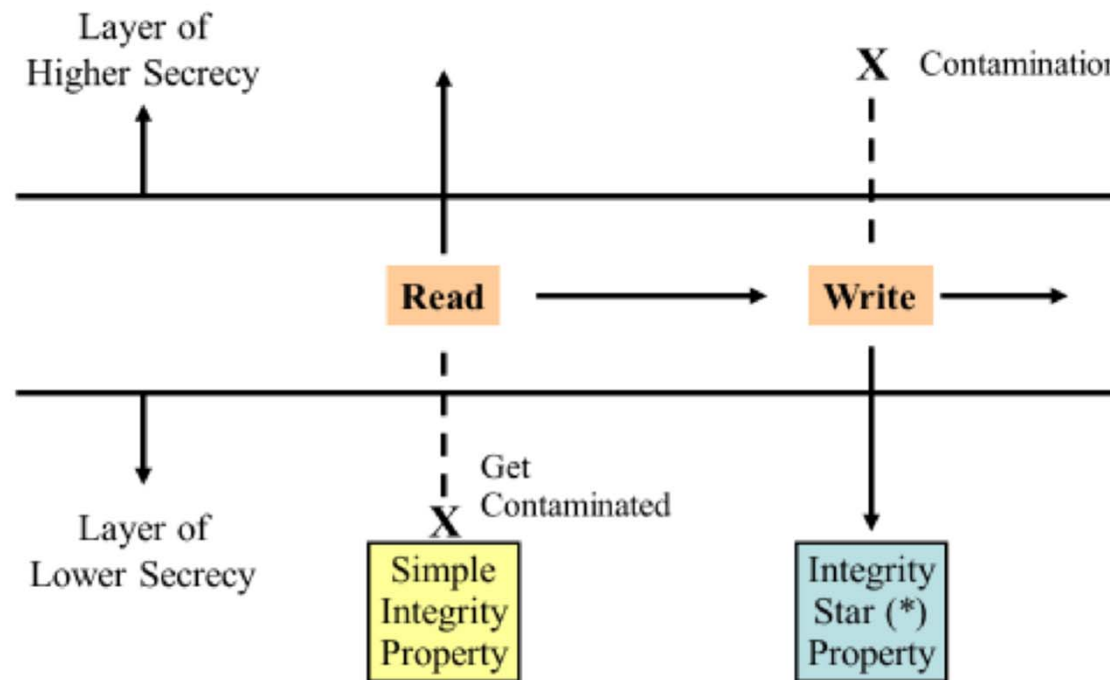
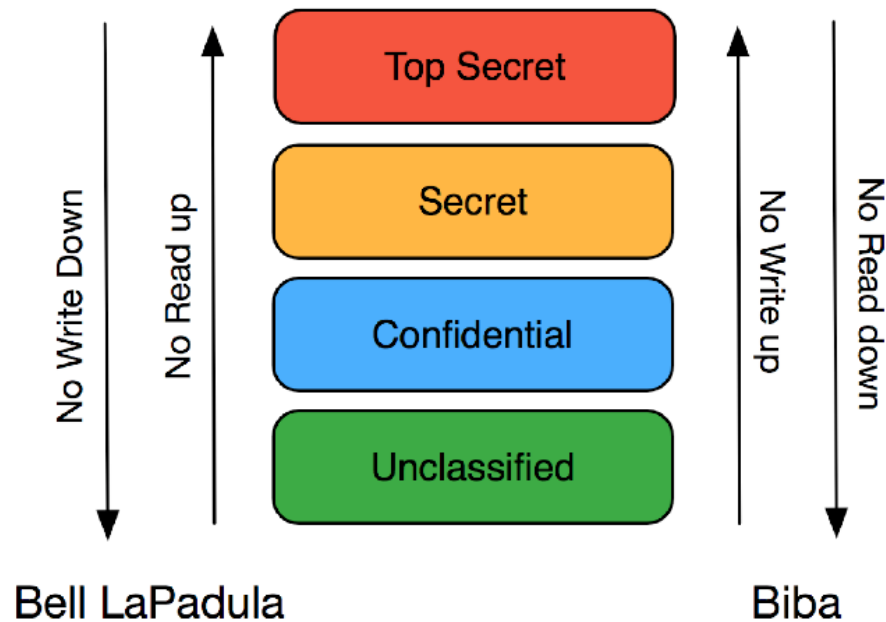


그림. BIBA 모델의 두 가지 주요 규칙

➤ BLP 모델과 BIBA 모델



Bell-LaPadula and Biba models.

- Bell-LaPadula provides confidentiality with the "no read up" and the "no write down" properties
- Biba provides integrity with the "no write up" and the "no read down"

➤ Reference Monitor (참조 모니터)

• 참조 모니터

- ✓ 주체의 객체로의 접근 허용 여부를 결정(접근 검사, 특수권한 관리, 보안 감사 메시지 생성 등)하는 추상적인 기계이다.
- ✓ 이것은 시스템의 보안 모델(예: BLP 또는 BIBA)을 강제하는 역할을 한다.
 - 예: 임의적 접근 제어 시스템에서의 참조 모니터는 보통 사용자가 패스워드 파일과 같은 제한된 파일에 쓰기를 방지
 - 예: 강제적 접근 제어 시스템에서의 참조 모니터는 Secret Subject가 Top Secret Object를 읽는 것을 방지
- ✓ 주체의 객체로의 접근 요청 및 허용 과정은 기록되어 나중에 감사 추적에 사용될 수 있다.

• 참조 모니터 요구 사항

- ✓ 참조 모니터 개념을 수행하는 프로세스는 독립적으로 분리되고(isolated), 부정하게 조작될 수 없어야(tamperproof) 한다.
- ✓ 참조 모니터는 모든 접근 시도에 대해 시행되어야 하고 회피하는 것이 불가능해야 한다.
- ✓ 모든 동작이 항상 분석과 테스트를 통해 올바름을 확인할 수 있어야 한다.

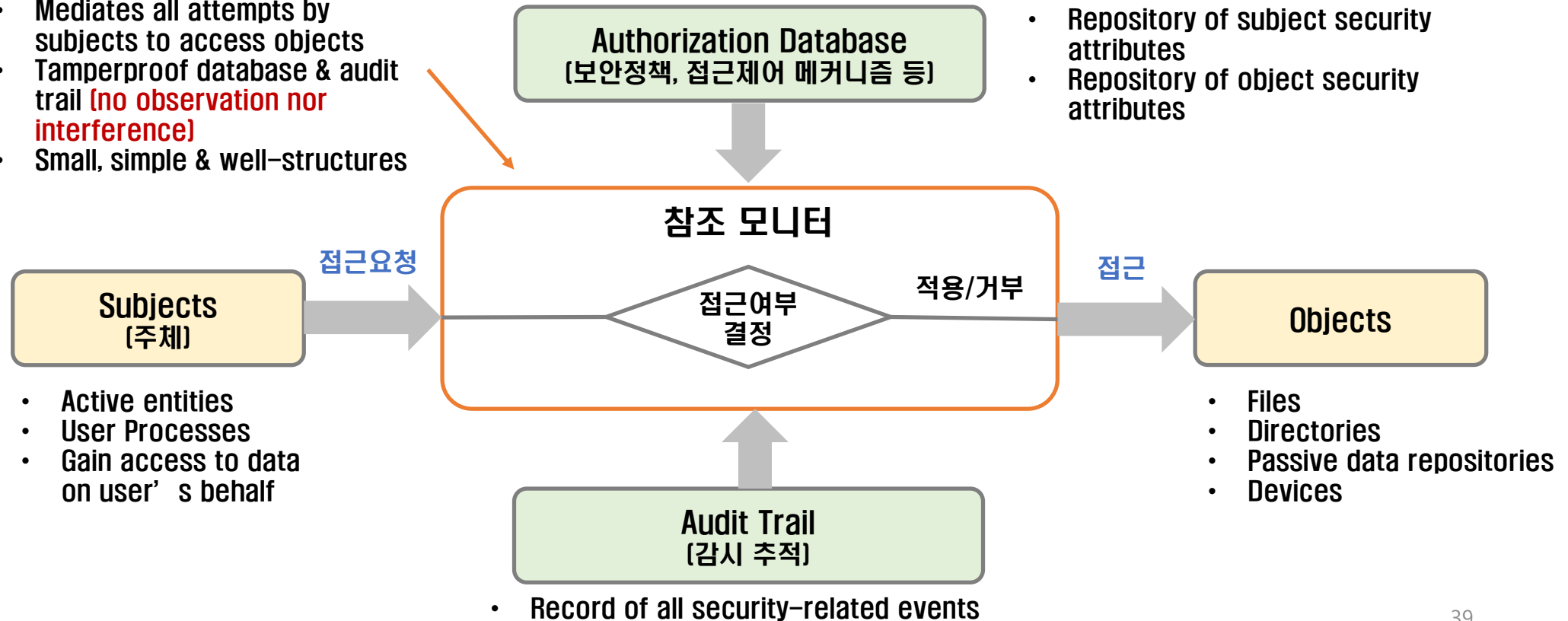
➤ Reference Monitor (참조 모니터)

- 보안 커널 및 보안 운영체제

- ✓ 참조 모니터를 구현한 것을 **보안 커널**이라 부르며, 이러한 보안 커널을 갖고 있는 운영체제를 **보안 운영체제**라 부른다.
- ✓ 보안 운영체제 환경에서 사용자 프로세스의 요청은 보안 검증을 위해서 반드시 참조 모니터를 거치도록 되어 있다.

➤ Reference Monitor (참조 모니터)

- Enforces security policy
- Mediates all attempts by subjects to access objects
- Tamperproof database & audit trail (**no observation nor interference**)
- Small, simple & well-structures



Access Control Methodology

- **종류**
 - 1) 중앙 집중형 접근 제어
 - 2) 분산형 접근 제어
 - 3) 혼합형 접근 제어
- **중앙 집중형 접근 제어**
 - ✓ 단일 개체가 전체 조직의 접근 제어를 수행하는 방식으로 AAA(Authentication, Authorization, Accountability) 기능을 하나의 시스템에서 수행한다.
 - ✓ AAA 프로토콜의 예: RADIUS, TACACS/TACACS+, Diameter

- 종류
 - 1) 중앙 집중형 접근 제어
 - 2) 분산형 접근 제어
 - 3) 혼합형 접근 제어
- 중앙 집중형 접근 제어
 - ✓ 단일 개체가 전체 조직의 접근 제어를 수행하는 방식으로 AAA(Authentication, Authorization, Accountability) 기능을 하나의 시스템에서 수행한다.
 - ✓ AAA 프로토콜의 예: RADIUS, TACACS/TACACS+, Diameter

- 분산형 접근 제어

- ✓ 각 자원의 사용자가 자신의 자원(예: 노트북)에 대해서 또는 한 부서의 부서장이 자신 부서의 시스템에 대해 인증, 인가, 책임 추적을 직접 제어하는 방식
- ✓ 중앙 집중형과 비교해볼 때, 해당 조직의 특성에 맞춰 빠르게 제어가 가능하다는 장점이 있지만, 일관된 제어 기법을 제공하기 어렵다는 단점이 있다.

- 혼합형 접근 제어

- ✓ 위 두 방식을 혼합한 형태이다.
- ✓ 예: 관리자는 중앙집중식으로 정보 시스템(데이터베이스, 파일 서버, 프린터, 호스트 등)에 대한 접근을 제어하며, 사용자는 분산 방식으로 자신 소유의 데이터에 대한 접근을 제어할 수 있다.

Access Control Applications

➤ OTP (One-Time Password)

• OTP 단말기

- ✓ 사용자가 사용할 때마다 매번 바뀌는 패스워드(보안 토큰)를 생성하는 장치
- ✓ OTP는 오직 한 번만 사용되는 패스워드로 다음과 같은 특성을 갖는다.
 - ✓ **패스워드의 재사용이 불가능**한 특징을 가지므로 추측을 통한 해킹이나 패킷 스니핑을 통한 재사용 공격이 불가능하며,
 - ✓ **역변환이 거의 불가능한 암호학적 알고리즘(예: 해쉬 함수)을 사용하여** 다음에 사용될 패스워드에 대한 예측 역시 불가능하다.

• OTP 단말기 분류

- ✓ 동기화 방식
 - 시간 동기화 방식
 - 이벤트 동기화 방식
- ✓ 비동기화 방식

OTP algorithm 찾기

- 시간 동기화 방식 OTP
 - ✓ OTP 단말기와 OTP 인증 서버 간에 시간값을 기준으로 동기화
 - ✓ 단말기 발급시 저장된 공유 비밀값과 동기화된 시간값을 해쉬 함수에 적용하여 OTP 생성
 - ✓ 일정 시간 범위 오차(30초~1분)를 허용함으로써 인증 허용
 - ✓ 대부분의 경우 이 방식을 사용함
- 이벤트 동기화 방식 OTP
 - ✓ 사용자가 OTP 단말기의 버튼을 누를 때마다 OTP 단말기 내부의 카운터값을 증가시킴
 - ✓ 단말기 발급시 저장된 공유 비밀값과 동기화된 카운터값을 해쉬 함수에 적용하여 OTP 생성
 - ✓ 인증할 때마다 인증 서버의 카운터도 동기화되어 증가됨

- 비동기화 방식 OTP

- ✓ **질의-응답(Challenge-Response) 방식을 사용하여 사용자를 인증한다.**
 - 인증 서버가 사용자에게 임의의 난수[질의]를 전달하면,
 - 사용자는 이를 OTP 장치에 직접 입력하며,
 - OTP 장치는 수신한 난수와 내장된 비밀값에 해쉬 함수를 적용하여 OTP(응답)를 출력한다.
- ✓ **사용자가 이 값(OTP)을 인증 서버로 전송하면, 서버는 동일한 방식으로 해당값을 만들고 이 값이 사용자가 보내온 값과 일치할 경우 인증이 완료된다.**
- ✓ 단말과 서버 간 동기화가 필요 없어 구조가 간단하나 사용자가 질의값을 직접 입력해야 하므로 사용이 번거롭다.
- ✓ 전수 조사 공격 또는 네트워크 모니터링에 의해 전송되는 값들이 노출될 경우 취약해진다.

➤ SSO (Single Sign On)

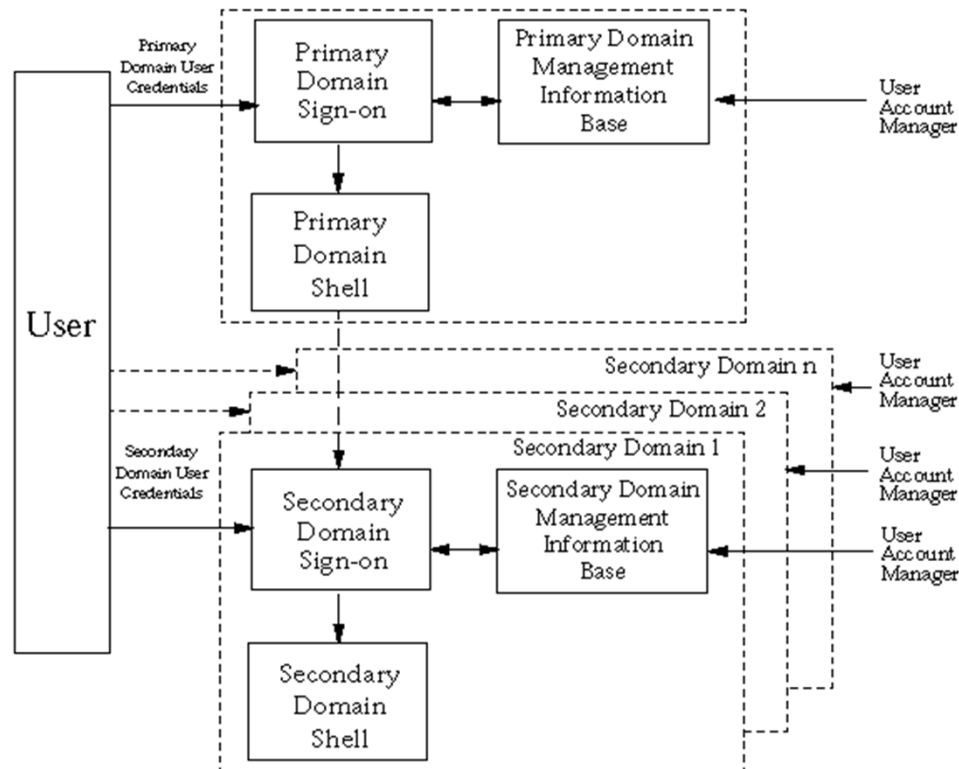
• SSO란

- ✓ 다양한 정보 시스템에 매번 아이디 및 패스워드를 입력하여 인증을 받지 않고, **한 번의 시스템 인증을 통하여 다양한 정보 시스템에 재인증 절차 없이 접근**하도록 하는 **통합 인증**을 말한다.
 - 1차 도메인(SSO 서버)에서 자격증명을 처리하고 이 정보를 2차 도메인(정보 자원, 응용 플랫폼)에 제공
 - 세션키를 나눠가지기 위해 커버로스, 세사미, 크립토나이트 등의 기술을 이용해 구현.
- ✓ 장점으로는 사용자가 시스템마다 별도의 패스워드를 관리할 필요없이 인증을 받을 수 있어 편의성이 증가하며, 중앙 집중 형태의 효율적인 관리가 가능해진다.
- ✓ 그러나, SSO 서버가 뚫리면 모든 서버의 침해가 가능하다는 단일 실패 지점 문제가 생긴다.

• SSO 구현 방안

- ✓ 웹 환경에서는 **쿠키 등을 활용**하여 **웹 시스템 SSO**를 구축하며,
- ✓ 웹 환경이 아닌 곳에서는 **ID/패스워드, 인증 토큰**(예: 커버로스) 등을 활용하여 구축할 수도 있다.

Legacy Approach to User Sign-on to Multiple Systems



Single User Sign-On To Multiple Services

