Chapter 3-2 : Access Control

사용자 인증, 접근 제어와 관련된 시스템

인증 → 보안 제 1 관문

접근제어 → 보안 제 2관문

Objectives

접근 제어 원칙 소개

- 주체(subjects), 객체(objects), 접근 권한(access right)

재량적 접근 제어 (주체가 객체에 대한 접근 권한 결정)

- 접근 행렬, 접근 제어 목록(ACLs), 기능 티켓(capability tickets)
- UNIX 전통적 및 ACL 메커니즘

Role-based(역할 기반)접근 제어

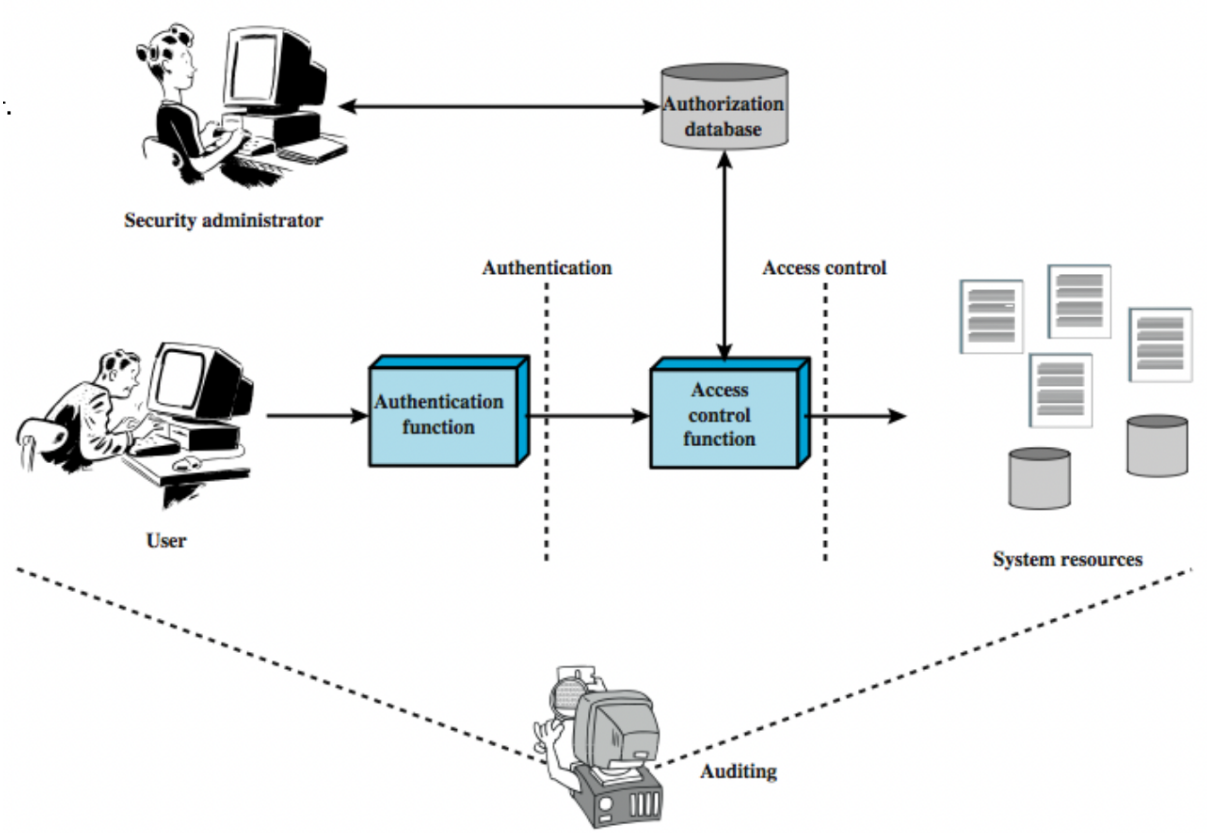
- Case Study

Attribute-based Access Control

- 속성기반 접근제어

Access Control

인증이 먼저 통과된 후, 사용자나 그룹이 시스템에 대한 인증이 완료된 상태인지를 확인 → 시스템 상의 어떤 자원에 대해 할당된 권한 / 기밀성에 대한 권한을 갖고 있다는 것을 기본으로 설계



- 자원의 부정적 사용을 막기 위해 만든 방식임.
  - 객체에 할당된 권한 == 주체에 허용된 권한인지 확인
- 컴퓨터 보안의 중심 요소
- 사용자와 그룹의 가정 :
  - 시스템에 인증(authentication)하고
  - 일부 시스템 자원에 대한 기밀성과 할당된 접근 권한을 갖고 있다
  - R, W, E 의 권한이 결정됨
- 접근 제어 원칙 (Access Control Principles)

Access Control Elements

- Subject (주체) : 객체에 접근 가능한 entity(user, app)
  - 사용자/응용프로그램을 대표하는 프로세스

    - 사용자 - 소유자(owner), 그룹(group), 전체 사용자(world)의 3개의 클래스를 갖는다
- Object (객체) : 보호해야 할 자원
    - Ex) 파일, 디렉토리, 레코드, 프로그램 등
    - 환경에 따라 개수/유형이 다양함
- Access right (접근 권한) : 주체가 객체에 접근하는 방식
    - Ex) read, write, execute, delete, create, search 등

Access Control Requirements

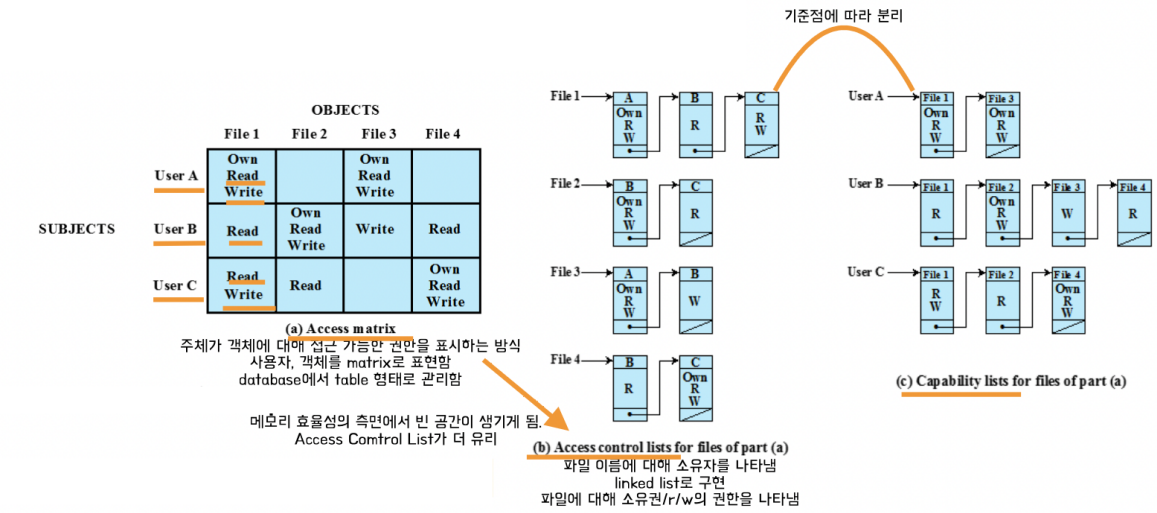
- 신뢰할 수 있는 입력 : 인증을 위한 메커니즘
- 상세/개략적 명세 : 다양한 수준(상, 중, 하 등 세분화)에서 접근을 규제(Ex - attribute, DB)
- Least Privilege : 업무 수행에 필요한 최소한의 권한만 부여함 (가장 기본)
- 업무 분리 : 각 단계를 서로 다른 사람들에게 분담
- Open/Closed 정책 : 특정 권한이 허용된 접근/금지된 접근을 제외한 모든 접근이 허용되는 것들 (다 막고 허용을 해주는 형태로 .. 정책)
- 관리 정책 : 규칙 추가, 삭제, 수정할 수 있는 사용자나 그룹 정하기
    - 누가 수정할 수 있는지에 대한 requirements가 명확히 정리되어 있을 때 access control이 되어야 한다

⭐Access Control Policies⭐

- ⭐**Mandatory Access Control, MAC**⭐
    - ***관리자가 정책에 근거한 접근 제어***
    - 강제 접근 제어
    - **security labels**(보안 레이블)과 **security clearances**(보안 승인)을 비교하는 것 기반

    - **Mandantory** - 자원에 접근하는 주체가 다른 주체에게 접근 권한 양보할 수 없음 (리눅스와 별개. 중앙 시스템 admit만이 이를 관리할 수 있음)
    - 중앙 집중식 보안 관리 - rule은 보안 전문가에 의해 생성되어, 운영자가 설정하고 OS에 의해 실행
    - Rule-Based 접근 제어 (규칙 기반 접근 제어)
    - Ex 1 - 네이버 카페
        - 가입 시 A, B, C 등의 등급별 권한이 정해짐. 운영자에 의해 엄격하게 관리된다.
        - 게시판(객체)에 할당된 등급에 따라 접근 권한이 제한된 거소가 같은 예시
    - Ex 2 - Network 방화벽
        - 미리 정해져 있는 rule에 기반함
        - 가장 적은 형태의 privilege를 기반으로 관리한다.
- **Discretionary Access Control, DAC**
    - ***신원, 사용자 중심의 접근 제어***
    - 자율 접근 제어 (임의의 접근 제어 권한. Ex - 773 등)
    - 주체가 속한 그룹의 신원에 따라 객체에 대한 접근을 제한하는 방법
    - 객체의 소유자가 접근 가능 여부를 결정함
- **Role-based Access Control, RBAC**
    - ***역할에 근거한 접근 제어***
    - 사용자 역할을 기반으로 함
    - Ex - 회사 조직에서 role을 기반으로 동작하는 방식임. A, B라는 사람의 권한을 중심으로, 사람이 바뀌면 접근제어 바뀜.
- **Attribute-based Access Control**
    - ***속성 기반 접근 제어***
    - 주체의 속성값(목소리 등), 객체의 속성값(파일이름, 크기, 생성일, 환경, 내부/외부 등)을 기반으로 접근 제어
    - But 정의해야 할 규칙 및 속성값이 너무 많음. 동작 환경도 고려하고, 과다 이용은 안되지만 개념은 좋다 → 클라우드 시스템 등에 도입되어 사용됨

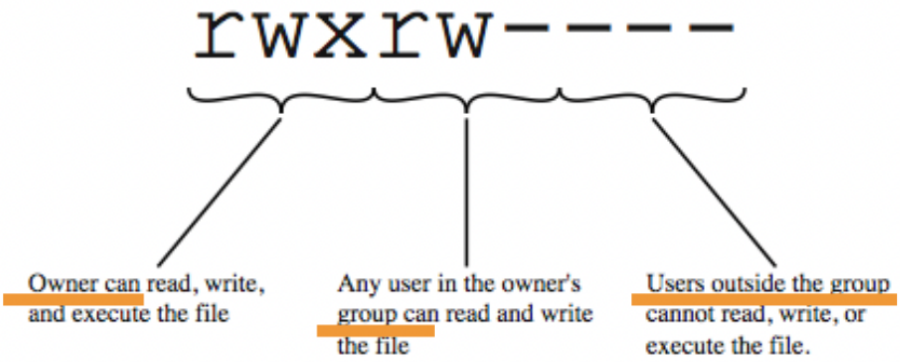
Discretionary Access Control



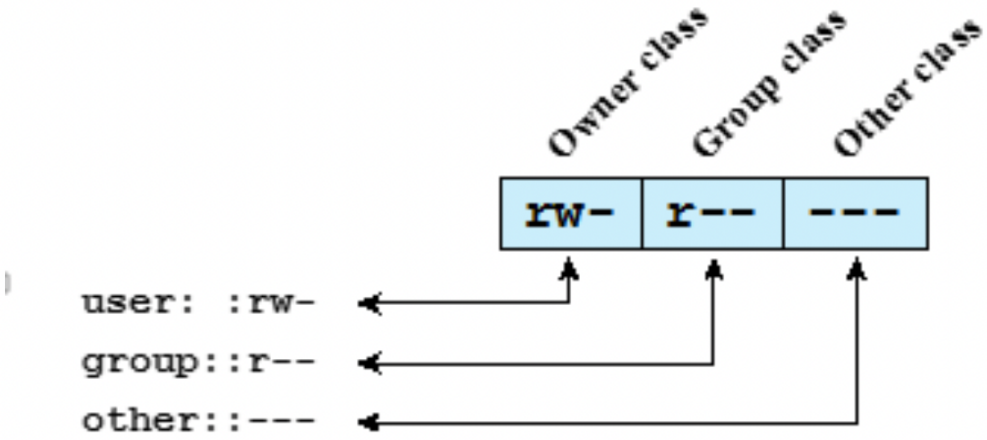
- 주로 Access Matrix를 사용해 제공됨
  - 각 항목은 지정된 주체의 해당 객체에 대한 access 권한을 지정함
  - row(행)에서 주체 나열 : **Capability tickets (lists)**
  - column(열)에서 객체 나열 : **Access control lists**
- 접근 행렬은 종종 빈 공간이 생기게 됨
  - 메모리의 효율성 측면에서 bad
- 행 또는 열 기준으로 분해 가능

UNIX 파일 접근제어

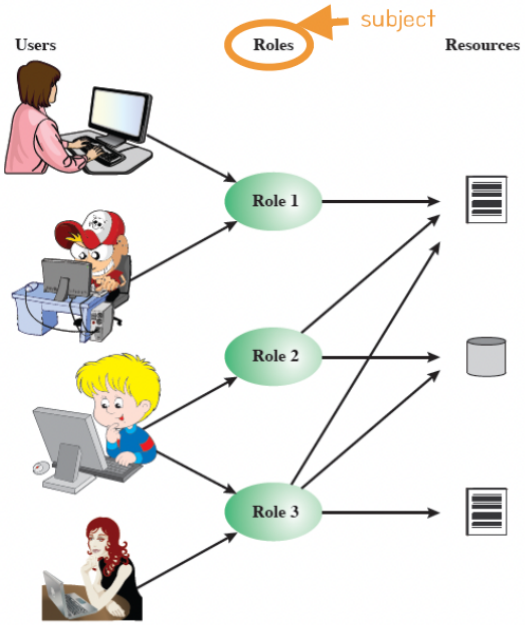
- 현대의 UNIX는 ACLs를 지원함
  - 추가 사용자/그룹 및 관련 rwx 권한을 지정할 수 있음
- UNIX 파일 접근제어
  - 12개 보호 비트



        - 9개 : 파일 소유자, 그룹 멤버 및 모든 다른 사용자의 읽기, 쓰기, 실행 권한 지정
        - 2개 : SetUID, SetGID를 지정
        - 1개 : Sticky bit (오직 소유자만 디렉토리를 제거, 삭제 등 가능)
    - Owner ID, Group ID, protection 비트는 파일의 inode 일부임
    - “set user ID(SetUID)”, “set Group ID(SetGID)”
        - 시스템이 일시적으로 오너와 그룹의 권한을 가짐. 효율성 고려해 임시적으로 오너의 권한을 갖도록 만듦. (공격의 빌미가 된다)
        - 특권 프로그램이 일반적으로 접근할 수 없는 파일
    - Sticky Bit
        - 디렉토리에서 소유자가 rename, move, delete를 제한함
- 객체에 대해 접근할 때
    - ACL 설정
    - 소유자, 관리자 등 나눔
    - 제대로 설정이 되었는지 검증

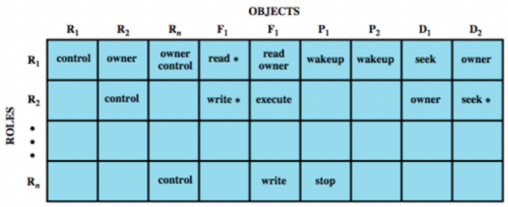


Role-Based 접근 제한



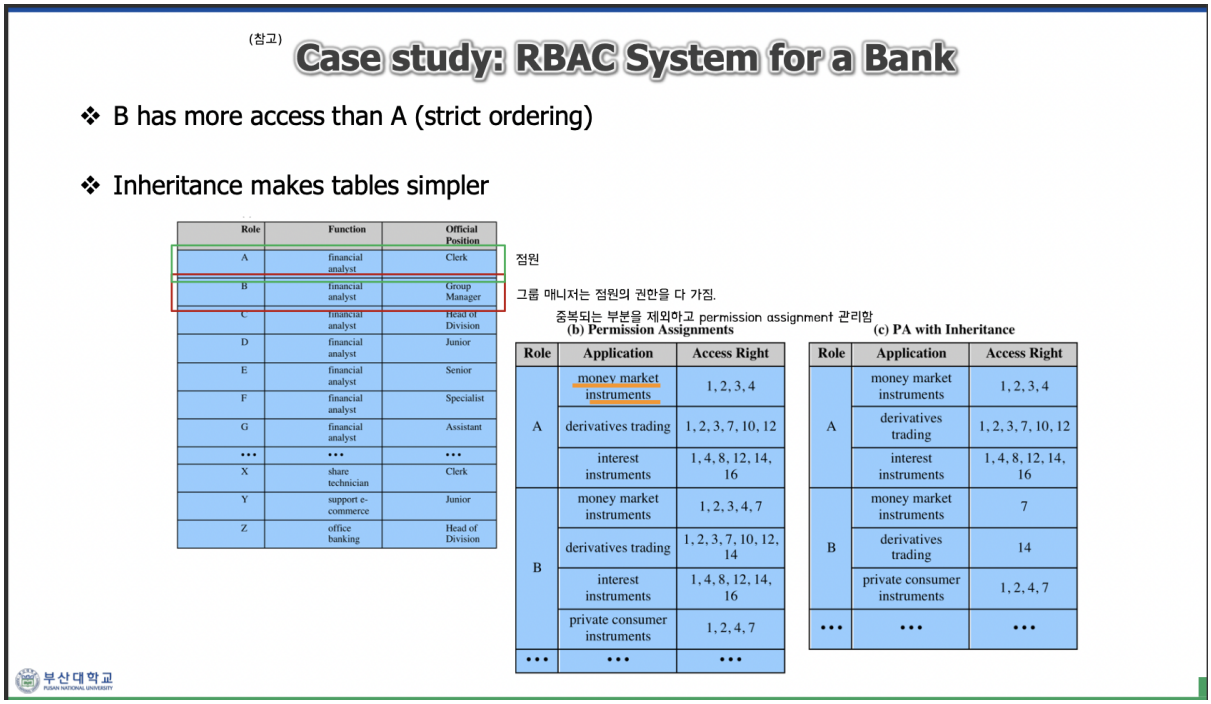
- '**Role**'(역할) 기반의 접근 제어 (identity 아님)
  - 1970년대의 다중 사용자, 다중 프로그래밍 환경에서 보안 처리 요구를 충족시키기 위해 제안된 방법
  - 사용자와 role간 **Many-to-many** 관계 (다대다). 하나의 role에 사용자 여럿이 대응될 수 있음. → 자원에 대한 접근 권한이 role 기준으로 이뤄짐.
  - role은 **static**(정적)임

- Role(역할)-Object(객체) 접근 행렬(matrix)



ROLES \ OBJECTS	R_1	R_2	R_n	F_1	F_1	P_1	P_2	D_1	D_2
R_1	control	owner	owner control	read *	read owner	wakeup	wakeup	seek	owner
R_2		control		write *	execute			owner	seek *
• • •									
R_n			control		write	stop			

Case Study : RBAC System for a Bank



(참고)

Case study: RBAC System for a Bank

❖ B has more access than A (strict ordering)

❖ Inheritance makes tables simpler

Role	Function	Official Position
A	financial analyst	Clerk
B	financial analyst	Group Manager
C	financial analyst	Head of Division
D	financial analyst	Junior
E	financial analyst	Senior
F	financial analyst	Specialist
G	financial analyst	Assistant
• • •	• • •	• • •
X	share technician	Clerk
Y	support e-commerce	Junior
Z	office banking	Head of Division

점원

그룹 매니저는 점원의 권한을 다 가짐.

중복되는 부분을 제외하고 permission assignment 관리함

(b) Permission Assignments

Role	Application	Access Right
A	money market instruments	1, 2, 3, 4
	derivatives trading	1, 2, 3, 7, 10, 12
	interest instruments	1, 4, 8, 12, 14, 16
B	money market instruments	1, 2, 3, 4, 7
	derivatives trading	1, 2, 3, 7, 10, 12, 14
	interest instruments	1, 4, 8, 12, 14, 16
	private consumer instruments	1, 2, 4, 7
• • •	• • •	• • •

(c) PA with Inheritance

Role	Application	Access Right
A	money market instruments	1, 2, 3, 4
	derivatives trading	1, 2, 3, 7, 10, 12
	interest instruments	1, 4, 8, 12, 14, 16
B	money market instruments	7
	derivatives trading	14
	private consumer instruments	1, 2, 4, 7
• • •	• • •	• • •

부산대학교 PUSAN NATIONAL UNIVERSITY

RBAC 모델

- Subject를 어디에 mapping 시키는지에 따라 분류함.
- **Role-based Model**
  - 회사에서 개개인의 역할에 기반해 접근제어하는 경우
  - 사용자가 특정 모드에서 정보 접근이 가능한 경우, 해당 사용자가 적절한 역할/접근 권한에 할당된 경우만 가능함
- **Task-based Model**

- 개개인의 업무에 따라 분류. 업무기반 통제

- **Lattice-based Model**
  - 격자를 통해 나눔 (민감도 레벨에 따라 더 세밀하게)
  - 상세히 내용기술이 가능하도록 나눈다.
  - task가 잘게 나눠진 것에 따라 보안등급이 나뉨
    - 주체, 객체에 보안 클래스를 부여

⭐MAC, DAC, RBAC 비교⭐

<요약> Comparison of MAC, DAC, RBAC

세 가지 구분하기

두 가지의 단점을 극복하기 위해
객체와 주체간의 역할 정의 방식에 따라 달라짐

Item	MAC	DAC	RBAC
Characteristics	Access control that gives access authority **by comparing the subject and object class** 정책에 기반해 mansatory하게 줌	Access control that gives access authority **according to the identity of the subject** 주체 (사용자)의 신원에 따라 권한 줌	A method of supplementing weakness of arbitrary and mandatory access control **by assigning role between subject and object**
Access Grant	System 시스템 관리자	Data owner 데이터 소유자	Central Authority 중앙 관리 (CEO에 의해)
Access Decision	Security Label	Identity	Role
Policy	엄격 Rigid 다른 사람에게 줄 수 없다	Flexible	Flexible
Advantage	Centralized, Stable 접근 제어만 생각했을 때, MAC이 가장 좋음	Flexible, Easy to implement 가용성이 좋음	Easy to manage 관리의 주체가 사람이 아님 !. 사람이 바뀌더라도 role만 관리
Disadvantage	Difficult to implement and manage, high cost 어려움	Vulnerable to Trojan horse, No way to control identity theft 신원, identity로 권한을 줄 수 있기 때문에 해킹될 수 있음 막는 방법이 없다. 악용가능성 ^	
Application	Firewall 방화벽	ACL	HIPAA (Health Insurance Probability and Accountability Act) 의사/간호사/수간호사 등을 관리 미국 의료관리

부산대학교 PUSAN NATIONAL UNIVERSITY 12

Attribute-based Access Control

- 자원, 주체 모두의 속성에 기반한 권한 제어 방식
- 여러가지 서비스를 운영할 때 제어하기 위한 방식. attribute에 따라 dimension이 커진다! → 속성을 어떻게 정의하는지 중요함

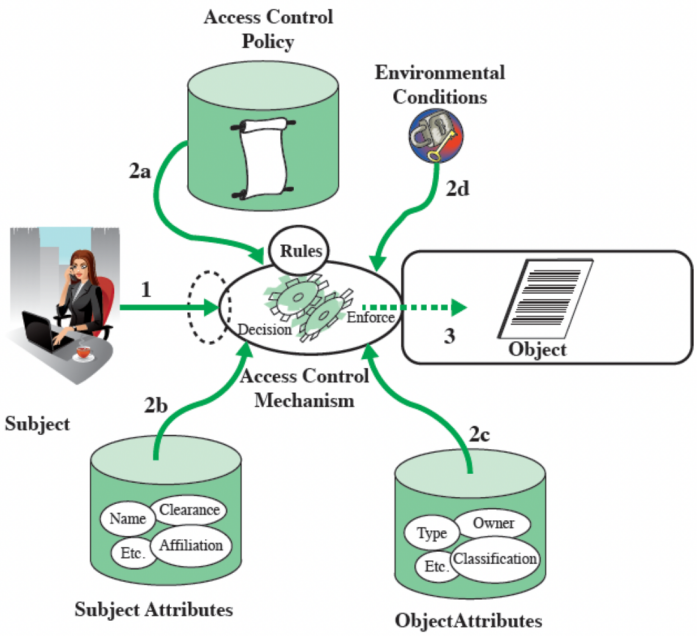
- 클라우드 서비스 모델

Types of Attributes

- **Subject attributes (주체 속성)**
  - 정보를 개체간 흐르게 하거나 시스템 상태를 변경하는 능동적 개체
  - attribute는 주체의 identity, characteristics를 정의 : **Name, Organization, Job title**
- **Object attributes (객체 속성, 파일 정의)**
  - 객체(자원)는 정보를 포함, 수신하는 수동적 정보 시스템 관련 개체
  - 객체는 접근 제어 결정을 내리는 데 활용될 수 있는 속성을 갖는다 : **Title, Author, Data**
- **Environment attributes (사용 환경 속성)**
  - 접근이 이뤄지는 운영, 기술, 상황적 환경, 문맥 의 요소 등 고려함
    - 현재 날짜
    - 현재 바이러스 / 해커 활동
    - 네트워크 보안 수준
    - 자원, 주체와 관련 없음
  - 대부분 접근 제어 정책에서 지금까지는 무시됨.

Sample ABAC Scenario

Attribute-based Access Control

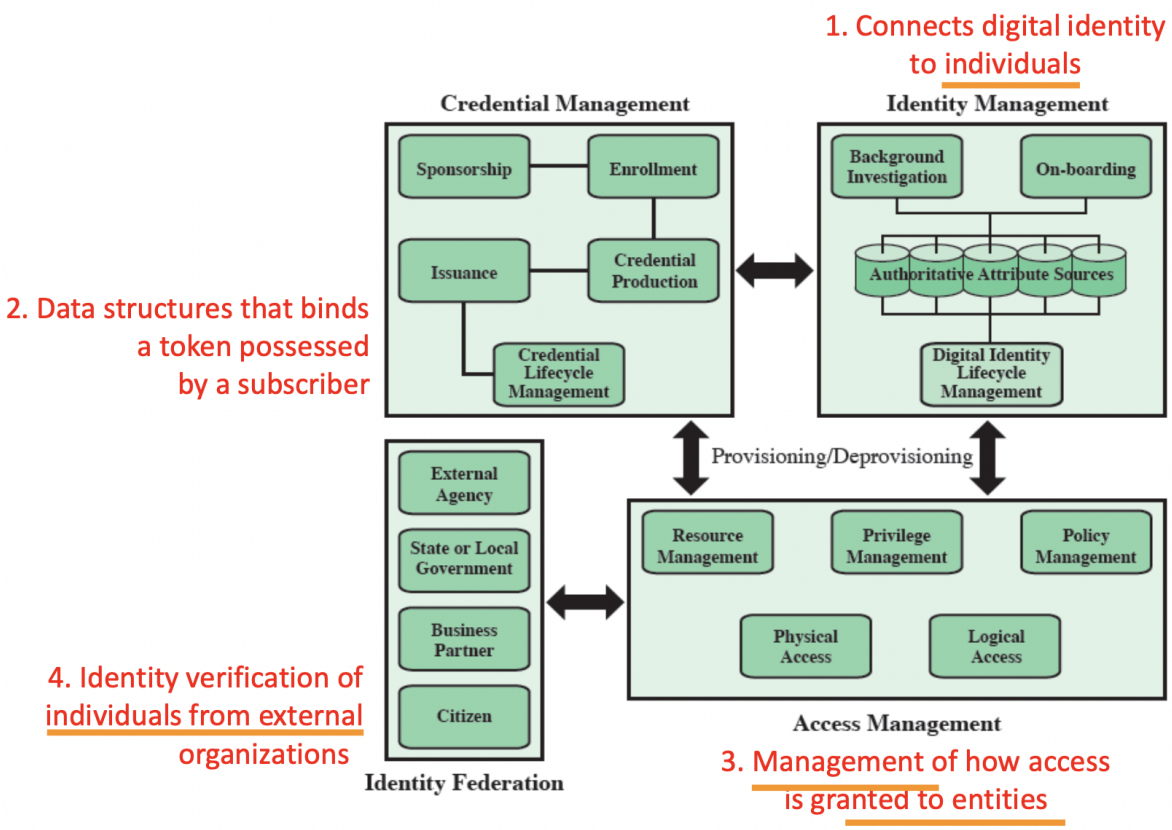


1. 주체가 객체에 대한 접근을 요청
2. AC(Access Control)는 규칙의 집합으로 제어됨. (2a) : 주체(2b), 객체(2c), 환경(2d)의 속성을 평가함
3. 권한이 부여되면 AC는 주체가 객체에 접근하도록 허용함

- 유연성을 제공하지만, 복잡도가 그만큼 커지게 됨!

ICAM

Identity, Credential, and Access Management



- 디지털 identities, credentials(자격 증명), access control을 관리 및 구현하는 종합적 접근 방법
- 개인 식별 시 신원을 기반으로 식별
  - identity → verification으로 변환함!
- 제로 트러스트 (아무것도 믿지 않음. 외부/내부 사용자를 모두 믿지 않고, 전방위적으로 막는 보안 방법)