CAN 304
Computer Systems Security

Lecture 7. Access Control

Week 8: 2020-04-15, 14:00-16:00, Friday

Jie Zhang
Department of Communications and Networking
Email: jie.zhang01@xjtlu.edu.cn
Office: EE522

Review of last time

- Introduction
- Basic authentication mechanisms
  - Something you know
  - Something you have
  - Biometric authentication

Learning objetives

- Define the three major categories of access control policies.
- Discuss the principal concepts of discretionary access control.
- Discuss the principal concepts of role-based access control.
- Discuss the principal concepts of attribute-based access control.

Outline

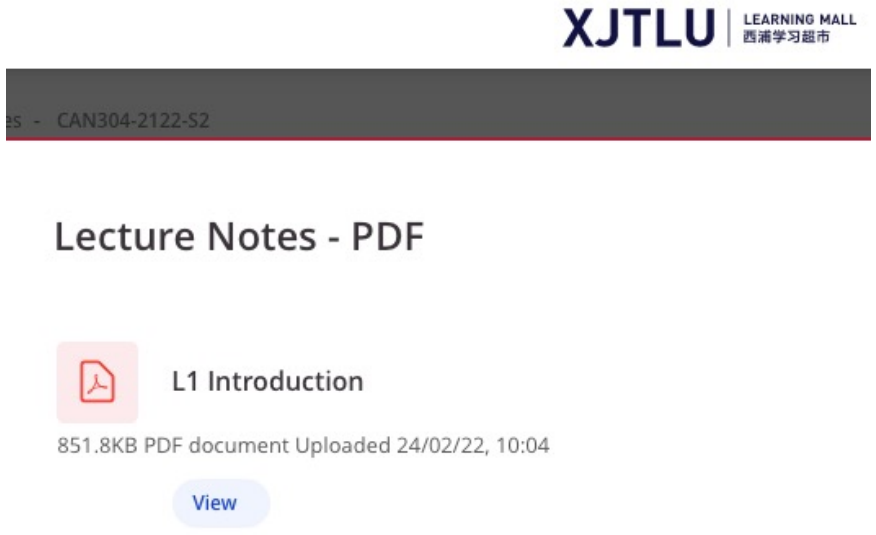
- Overview of concepts
- Access control policies and implementations

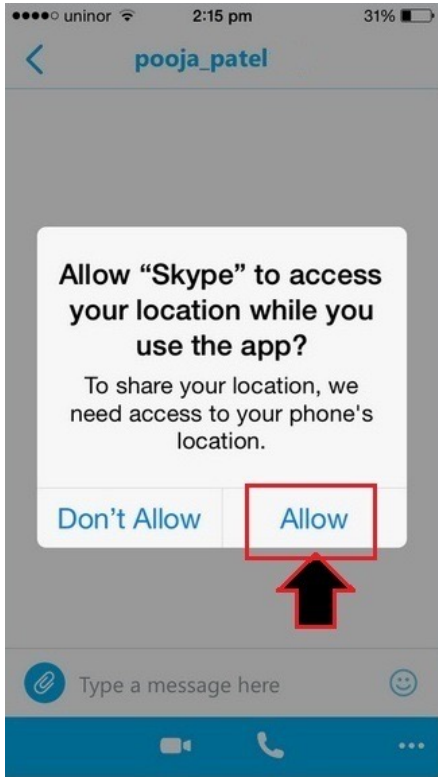
1. Overview of concepts

Access Control

RFC 4949 defines access control as:

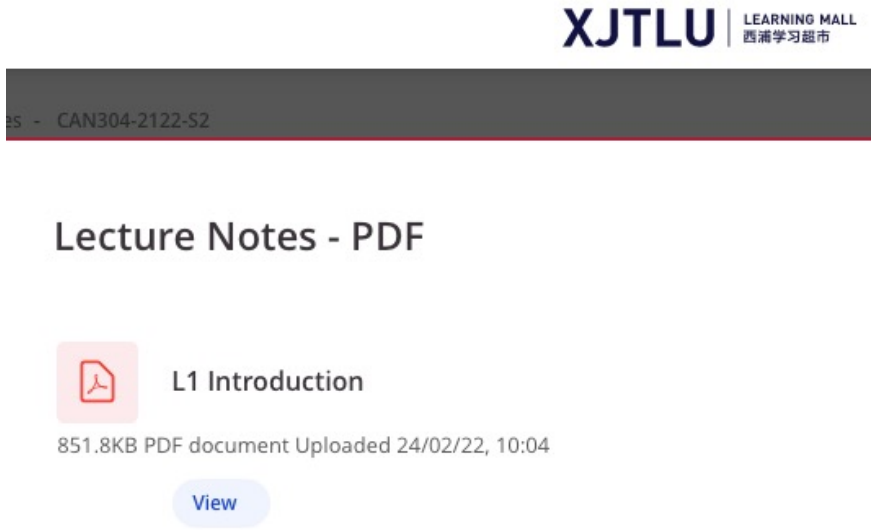
- “A process by which use of system resources is regulated according to a security policy and is permitted only by authorized entities (users, programs, process, or other systems) according to that policy”

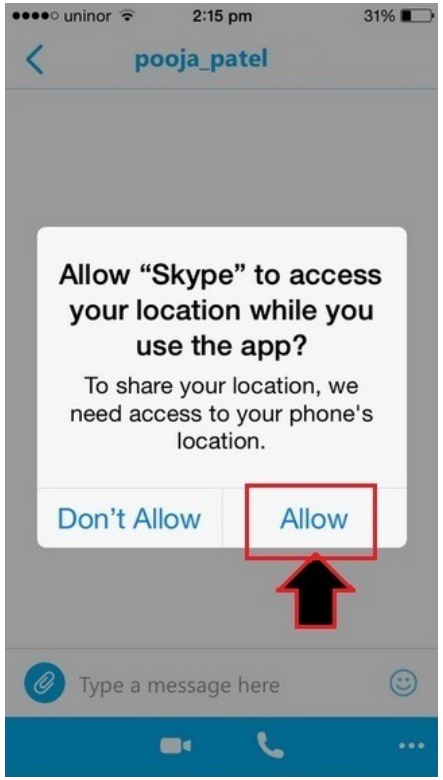




Access Control

Access control implements a security policy that specifies **who or what** (e.g., in the case of a process) may have access to each specific **system resource** and the **type of access** that is permitted in each instance.





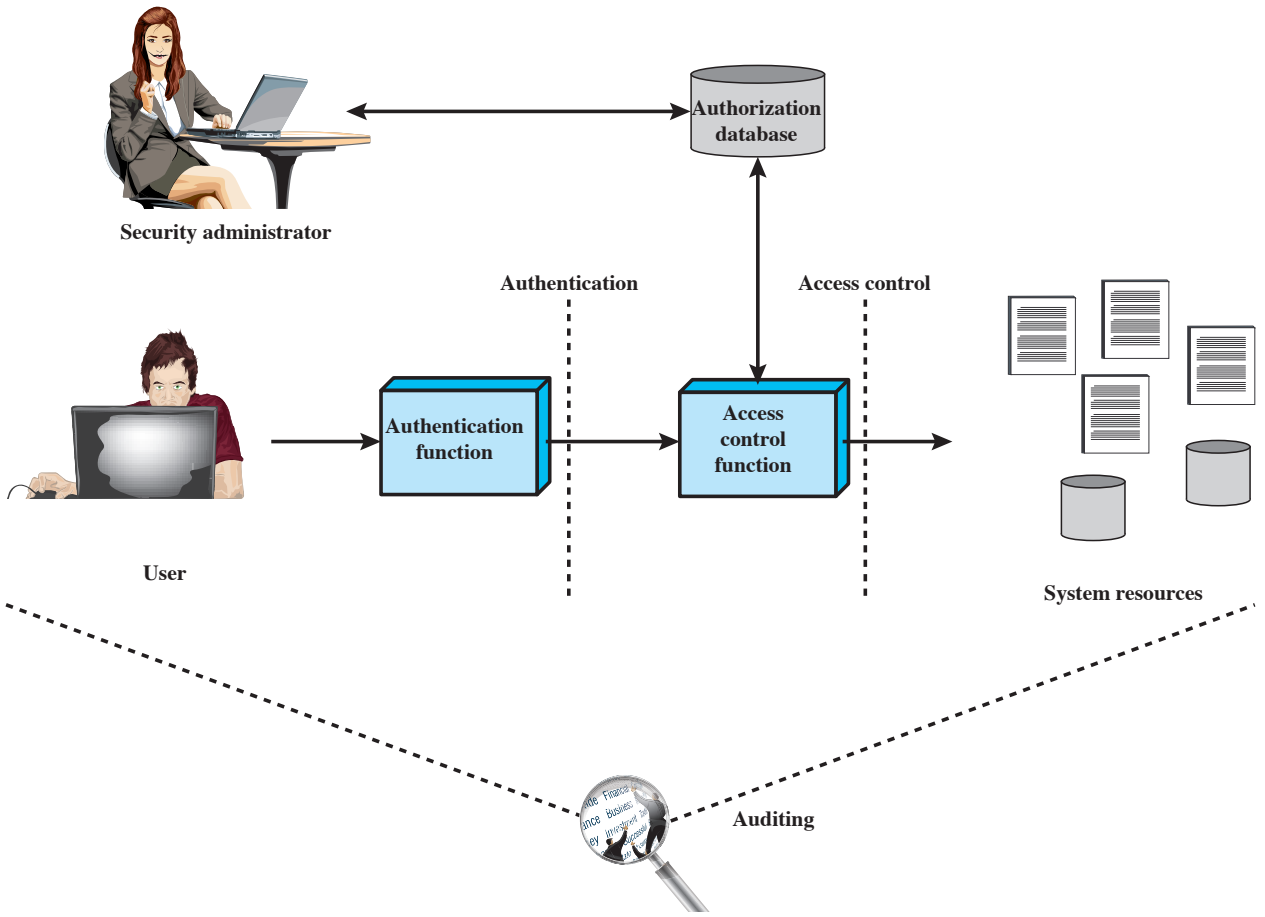


Figure 4.1 Relationship Among Access Control and Other Security Functions

Basic Elements of Access Control

Subject:

- An entity capable of accessing objects
- E.g., a user of learning mall

Object:

- A resource to which access is controlled
- E.g., a file, a coursework module, etc. on the module page in learning mall
- Object is an entity used to contain and/or receive information

Access right:

- Describes the way in which a subject may access an object
- Could include: Read, Write, Execute, Delete, Create, and Search

Access Control Policies

An access control policy dictates what types of access are permitted, under what circumstances, and by whom.

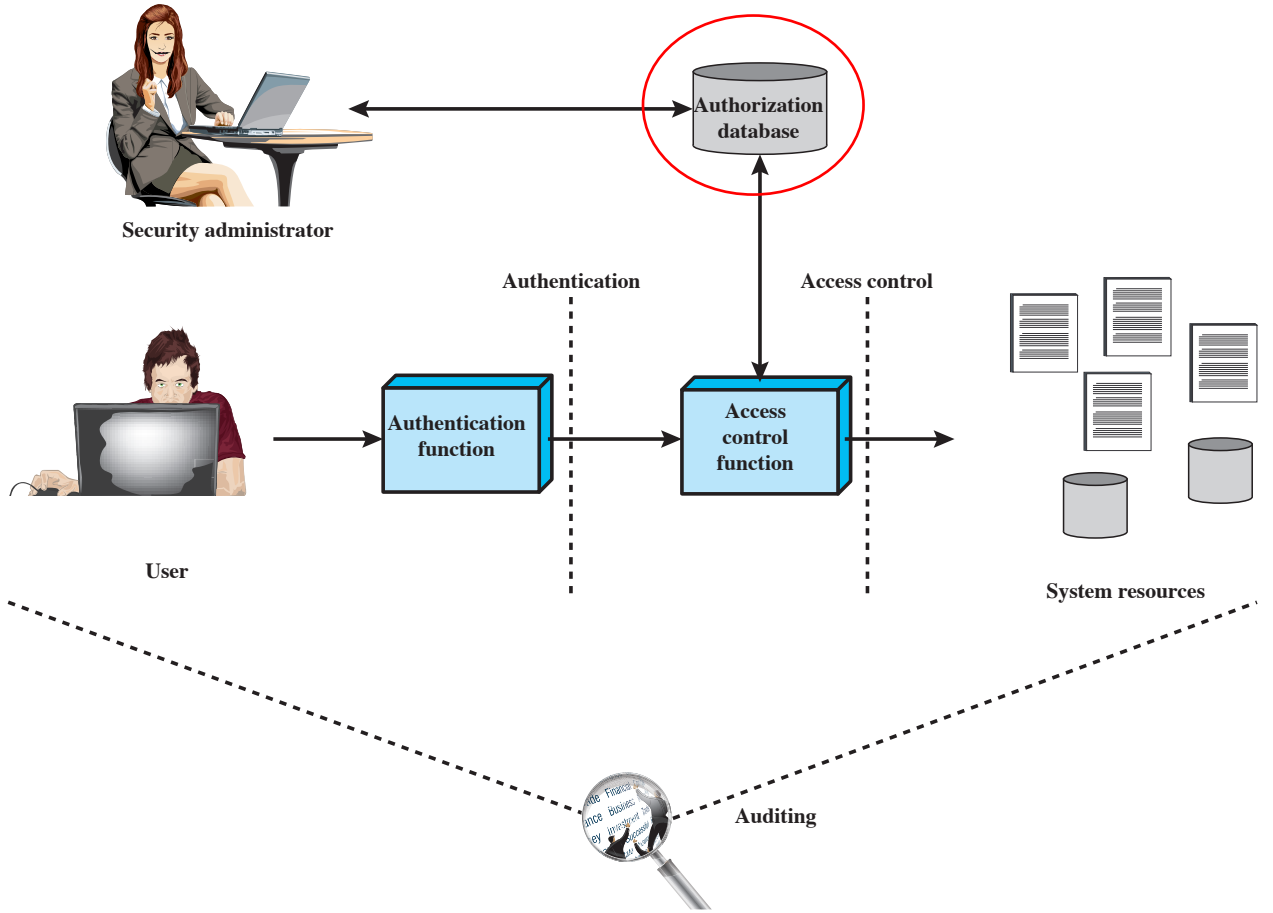


Figure 4.1 Relationship Among Access Control and Other Security Functions

Access Control Policies

Discretionary access control (DAC)

- Controls access based on the **identity** of the requestor and on access rules (authorizations) stating what requestors are (or are not) allowed to do.
- An entity might have access rights that permit the entity, by its own volition, to enable another entity to access some resource.

Mandatory access control (MAC)

- Controls access based on comparing security labels with **security clearances**
- **Centrally controlled**, e.g., user cannot grant access
- For military information security

Role-based access control (RBAC)

- Controls access based on the **roles** that users have within the system and on rules stating what accesses are allowed to users in given roles.

Attribute-based access control (ABAC)

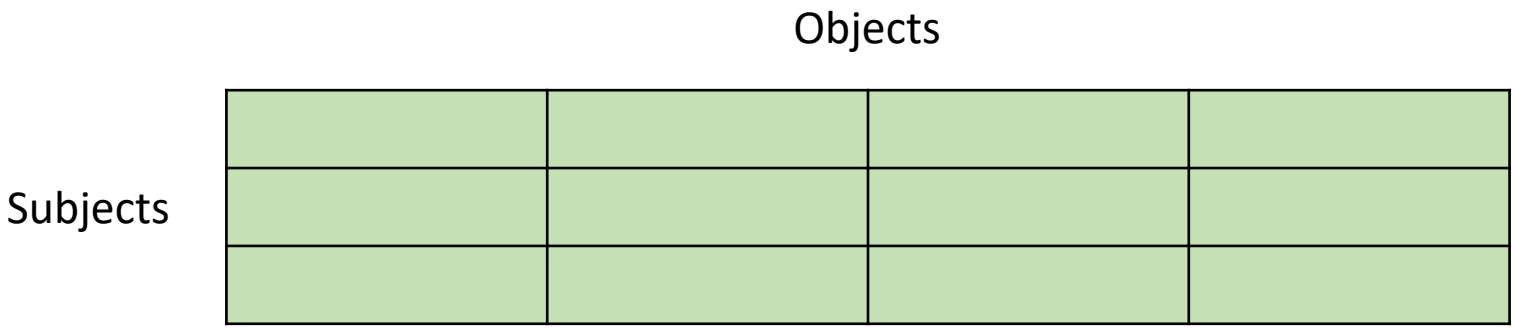
- Controls access based on **attributes** of the user, the resource to be accessed, and current environmental conditions

2. Access control policies and implementations

2.1 Discretionary access control

Discretionary Access Control (DAC)

- Scheme in which an entity may enable another entity to access some resource
- Often provided using an access matrix
  - One dimension consists of identified subjects that may attempt data access to the resources
  - The other dimension lists the objects that may be accessed
  - Each entry in the matrix indicates the access rights of a particular subject for a particular object

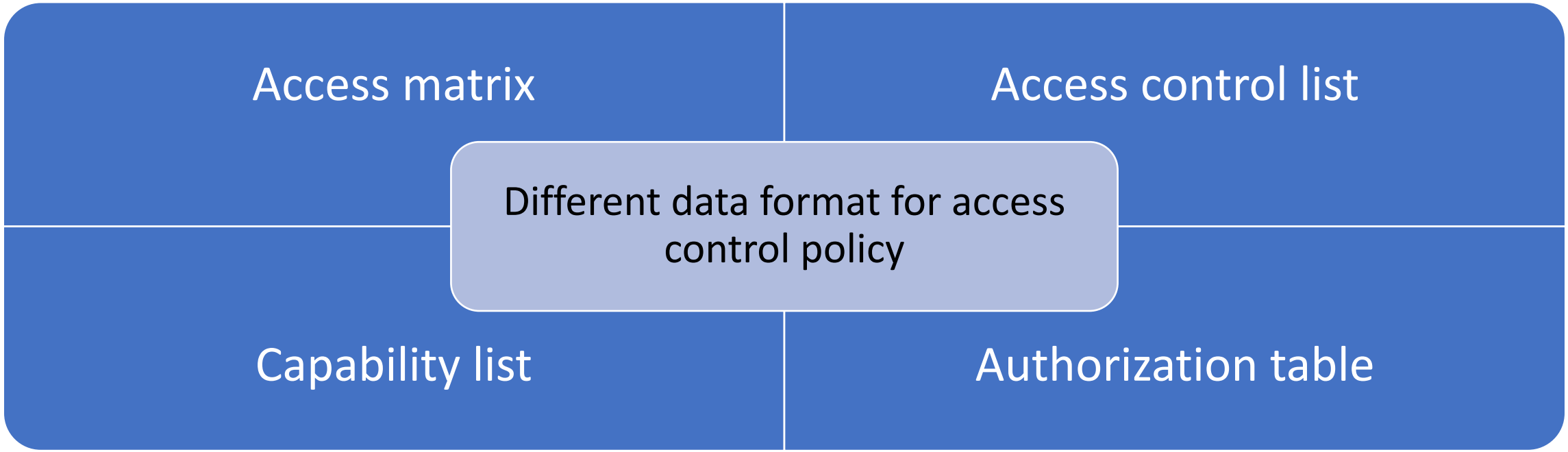


Access matrix

		OBJECTS			
		File 1	File 2	File 3	File 4
SUBJECTS	User A	Own Read Write		Own Read Write	
	User B	Read	Own Read Write	Write	Read
	User C	Read Write	Read		Own Read Write

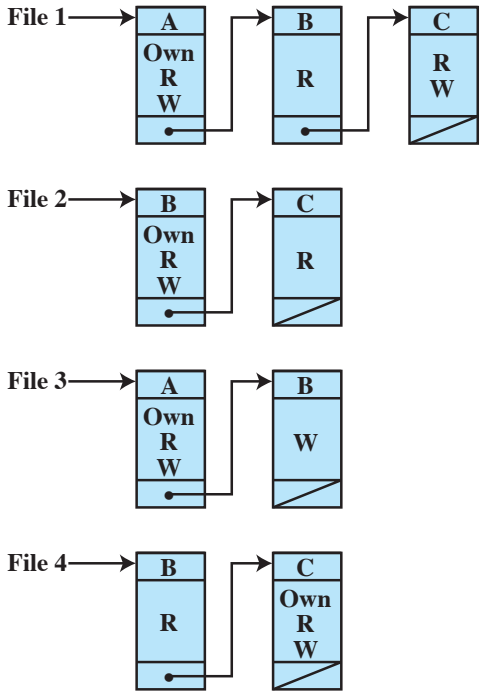
(a) Access matrix

Access Matrix and Other Access Control Data Structure



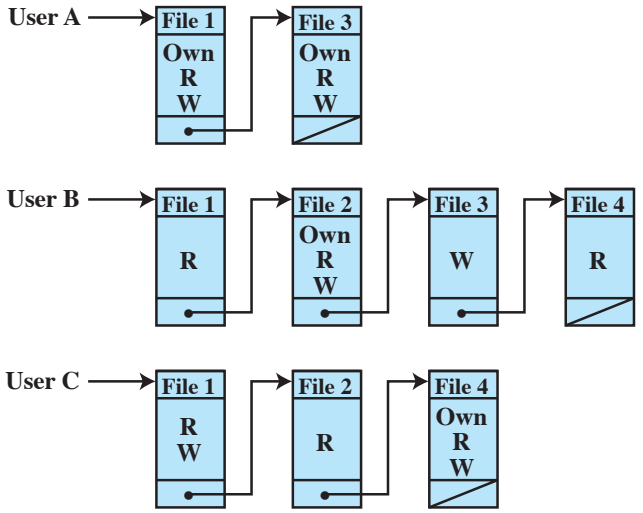
Access control list & Capability list

By objects



(b) Access control lists for files of part (a)

By subjects



(c) Capability lists for files of part (a)

OBJECTS

SUBJECTS	File 1	File 2	File 3	File 4
User A	Own Read Write		Own Read Write	
User B	Read	Own Read Write	Write	Read
User C	Read Write	Read		Own Read Write

(a) Access matrix

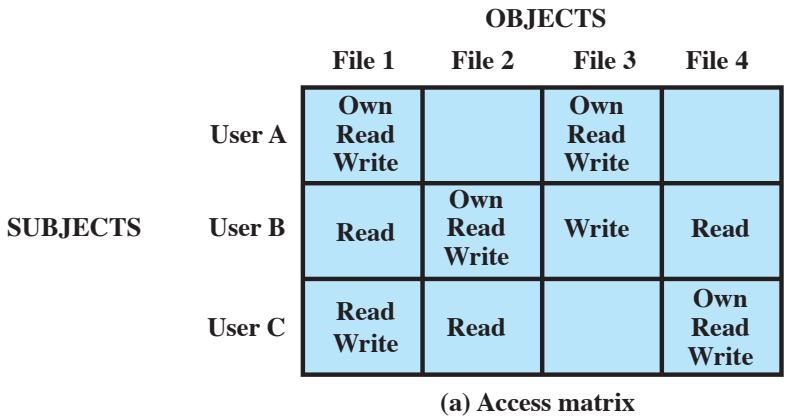
To determine the access rights available to a specific user.

To determine which subjects have which access rights to a particular resource.

Figure 4.2 Example of Access Control Structures

Authorization table

Subject	Access Mode	Object
A	Own	File 1
A	Read	File 1
A	Write	File 1
A	Own	File 3
A	Read	File 3
A	Write	File 3
B	Read	File 1
B	Own	File 2
B	Read	File 2
B	Write	File 2
B	Write	File 3
B	Read	File 4
C	Read	File 1
C	Write	File 1
C	Read	File 2
C	Own	File 4
C	Read	File 4
C	Write	File 4



(a) Access matrix

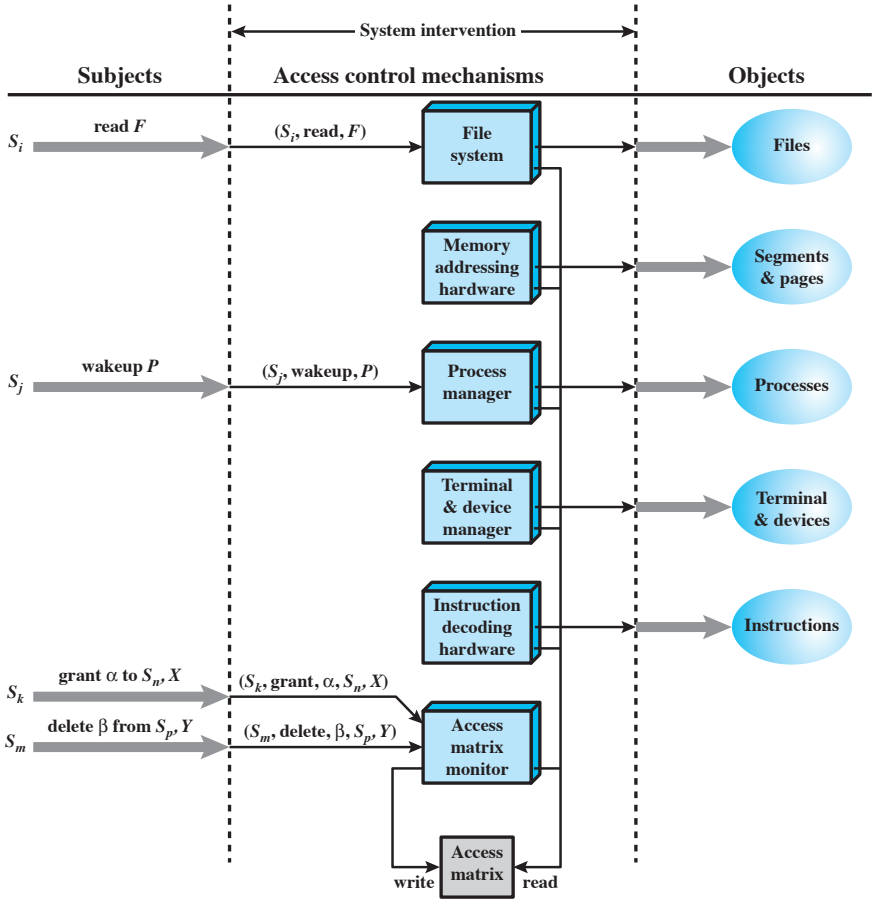
A general model for DAC

- The model assumes a set of subjects, a set of objects, and a set of rules that govern the access of subjects to objects.

		OBJECTS								
		subjects			files		processes		disk drives	
		S_1	S_2	S_3	F_1	F_2	P_1	P_2	D_1	D_2
SUBJECTS	S_1	control	owner	owner control	read *	read owner	wakeup	wakeup	seek	owner
	S_2		control		write *	execute			owner	seek *
	S_3			control		write	stop			

* - copy flag set

Figure 4.3 Extended Access Control Matrix



		OBJECTS								
		subjects			files		processes		disk drives	
		S_1	S_2	S_3	F_1	F_2	P_1	P_2	D_1	D_2
SUBJECTS	S_1	control	owner	owner control	read *	read owner	wakeup	wakeup	seek	owner
	S_2		control		write *	execute			owner	seek *
	S_3			control		write	stop			

		S_1	S_2	S_3	F_1	F_2	P_1	P_2	D_1	D_2
SUBJECTS	S_1	control	owner	owner control	read *	read owner	wakeup	wakeup	seek	owner
	S_2		control		write *	execute			owner	seek *
	S_3			control		write	stop			

Rule	Command (by S_o)	Authorization	Operation
R1	**transfer** $\begin{Bmatrix}\alpha* \\ \alpha\end{Bmatrix}$ **to** S, X	'α*' in $A[S_o, X]$	store $\begin{Bmatrix}\alpha* \\ \alpha\end{Bmatrix}$ in $A[S, X]$
R2	**grant** $\begin{Bmatrix}\alpha* \\ \alpha\end{Bmatrix}$ **to** S, X	'owner' in $A[S_o, X]$	store $\begin{Bmatrix}\alpha* \\ \alpha\end{Bmatrix}$ in $A[S, X]$
R3	**delete** α **from** S, X	'control' in $A[S_o, S]$ or 'owner' in $A[S_o, X]$	delete α from $A[S, X]$
R4	$w \leftarrow$ **read** S, X	'control' in $A[S_o, S]$ or 'owner' in $A[S_o, X]$	copy $A[S, X]$ into w
R5	**create object** X	None	add column for X to A; store 'owner' in $A[S_o, X]$
R6	**destroy object** X	'owner' in $A[S_o, X]$	delete column for X from A
R7	**create subject** S	none	add row for S to A; execute **create object** S; store 'control' in $A[S, S]$
R8	**destroy subject** S	'owner' in $A[S_o, S]$	delete row for S from A; execute **destroy object** S

Table 4.2

Access Control System Commands

Protection Domains

- A set of objects together with access rights to those objects
- In terms of the access matrix, a row defines a protection domain

		OBJECTS			
		File 1	File 2	File 3	File 4
SUBJECTS	User A	Own Read Write		Own Read Write	
	User B	Read	Own Read Write	Write	Read
	User C	Read Write	Read		Own Read Write

(a) Access matrix

- E.g., UNIX
  - In user mode certain areas of memory are protected from use and certain instructions may not be executed
  - In kernel mode privileged instructions may be executed and protected areas of memory may be accessed

2.2 Role-based access control

Role-Based Access Control (RBAC)

- Controls access based on the **roles** that users have within the system and on rules stating what accesses are allowed to users in given roles
- E.g., Job function within an organization

- In contrast, DAC is based on user's **identity**

- Users are assigned to roles
- The relationship of users to roles is many to many, as is roles to resources/objects
- The user-to-role assignment can be dynamic

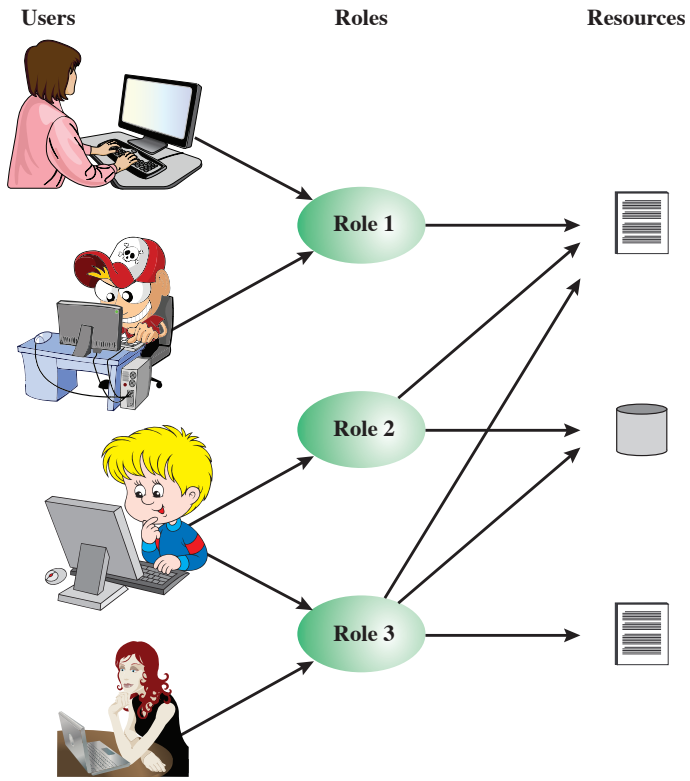
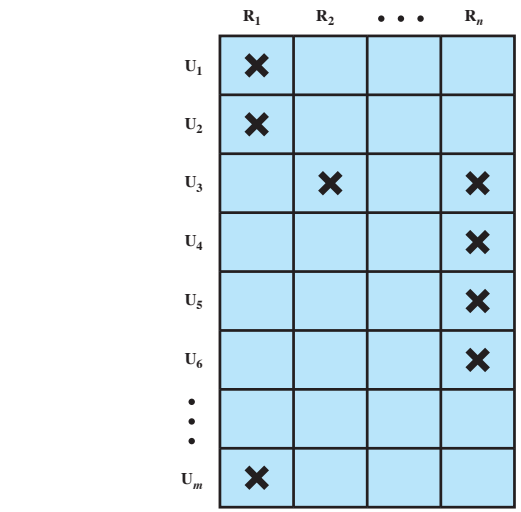


Figure 4.6 Users, Roles, and Resources



OBJECTS

ROLES	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			

Figure 4.7 Access Control Matrix Representation of RBAC

RBAC Reference Models

$RBAC_0$

- Contains the minimum functionality for an RBAC system

$RBAC_1$

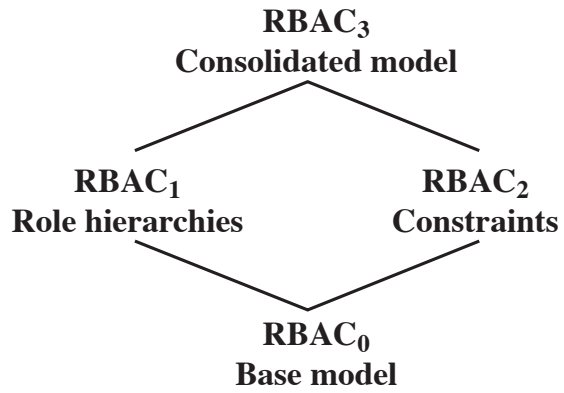
- $RBAC_0$ + role hierarchies

$RBAC_2$

- $RBAC_0$ + constraints

$RBAC_3$

- $RBAC_1$ + $RBAC_2$



(a) Relationship among RBAC models

Models	Hierarchies	Constraints
$RBAC_0$	No	No
$RBAC_1$	Yes	No
$RBAC_2$	No	Yes
$RBAC_3$	Yes	Yes

Table 4.3 Scope RBAC Models

Base Model—$RBAC_0$

User:

- An individual that has access to this computer system.

Role:

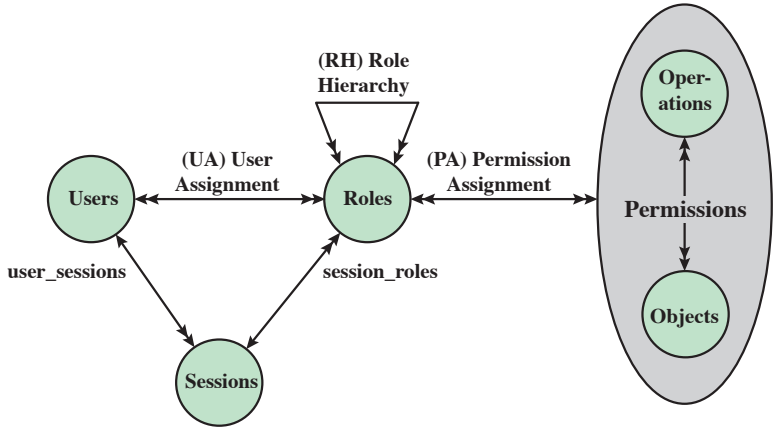
- A named job function within the organization that controls this computer system.

Permission:

- An approval of a particular mode of access to one or more objects.

Session:

- A mapping between a user and an activated subset of the set of roles to which the user is assigned.



(b) RBAC models

Figure 4.8 A Family of Role-Based Access Control Models.

Role Hierarchies—$RBAC_1$

- Role hierarchies provide a means of reflecting the hierarchical structure of roles in an organization.

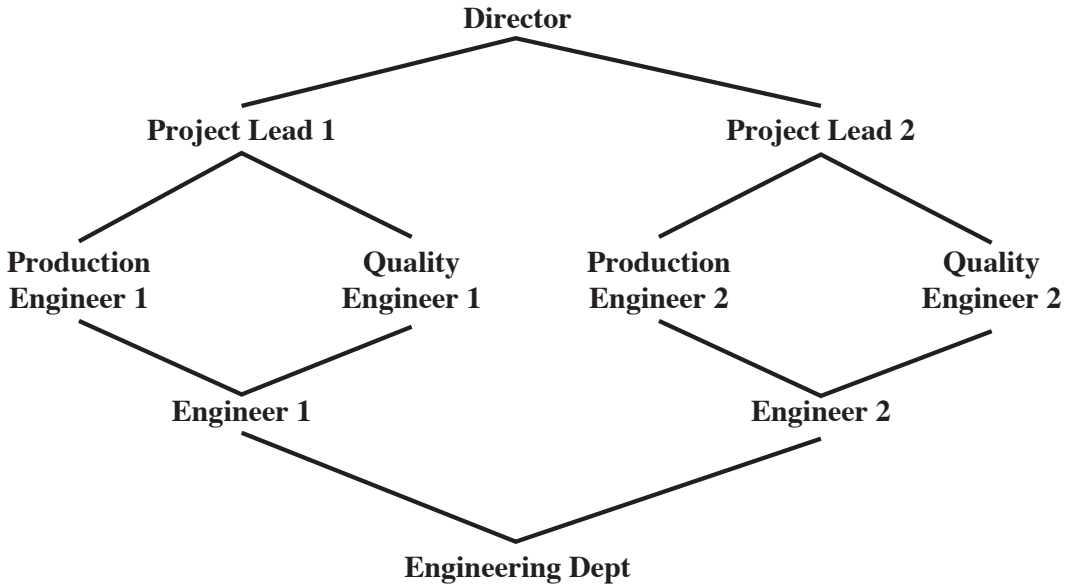
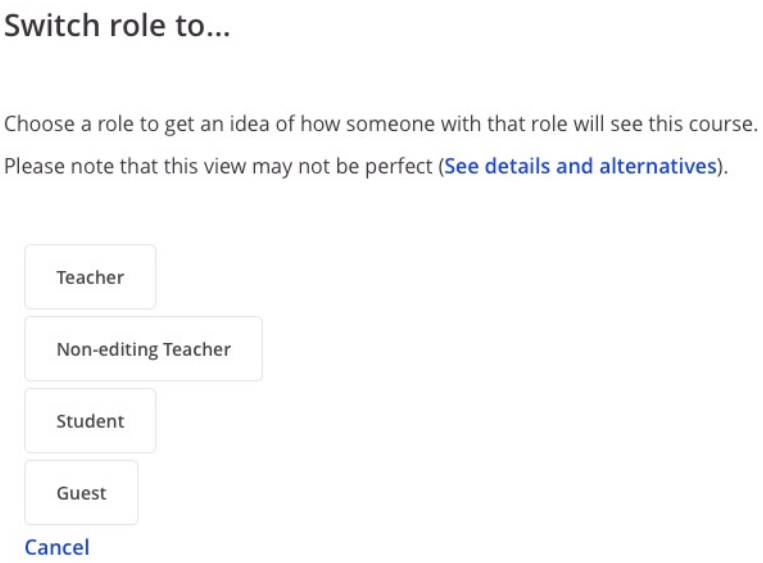


Figure 4.9 Example of Role Hierarchy



RBAC Constraints—$RBAC_2$

- Provide a means of adapting RBAC to the specifics of administrative and security policies of an organization
- A defined relationship among roles or a condition related to roles
- Types:

Mutually exclusive roles

- A user can only be assigned to one role in the set (either during a session or statically)
- Any permission (access right) can be granted to only one role in the set

Cardinality

- Setting a maximum number with respect to roles

Prerequisite roles

- Dictates that a user can only be assigned to a particular role if it is already assigned to some other specified role

2.3 Attribute-based access control

Attribute-Based Access Control (ABAC)

- A relatively recent development in access control technology.
- Can define authorizations that express conditions on properties of both the resource and the subject.
- Strength is its flexibility and expressive power.
- There are three key elements to an ABAC model:
  - attributes
  - policy model
  - architecture model

ABAC Model: Attributes

Subject attributes

- A subject, e.g., a user, an application, a process, or a device.
- Each subject has associated attributes that define the identity and characteristics of the subject, e.g., identifier, name, organization, job title

Object attributes

- An object (i.e., resource), e.g., devices, files, records, tables, processes, programs, networks, domains
- Objects have attributes that can be leveraged to make access control decisions, e.g., title, subject, date, and author for a MS word file

Environment attributes

- Have so far been largely ignored in most access control policies.
- Examples: current date and time, the current virus/hacker activities, and the network's security level (e.g., Internet vs. intranet)

ABAC Logical Architecture

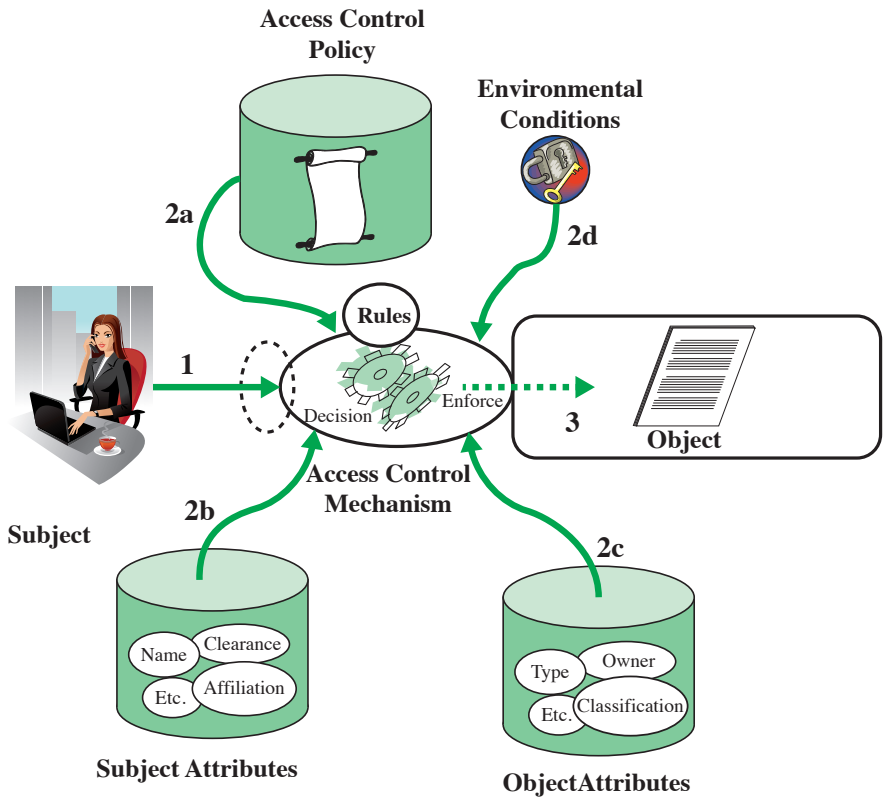


Figure 4.10 Simple ABAC Scenario

ABAC Policies

- A policy is **a set of rules and relationships** that govern allowable behavior within an organization, based on the **privileges** of subjects and how resources or objects are to be **protected** under which environment **conditions**.

- Typically written from the perspective of the object that needs protecting and the privileges available to subjects

An ABAC policy model

- S, O, E stand for subjects, objects, and environments respectively

- $SA_k\ (1 \leq k \leq K), OA_m\ (1 \leq m \leq M), EA_n\ (1 \leq n \leq N)$ are pre-defined attributes for subjects, objects, and environments, respectively

- $ATTR(s), ATTR(o), ATTR(e)$ are attribute assignment relations for subject s, object o, and environment e, respectively.
  - Also used for the value assignment of individual attributes. For example,

$$Role(s) = \text{“}Module\ Leader\text{”}$$
$$ServiceOwner(o) = \text{“}xjtlu\text{”}$$
$$CurrentDate(e) = \text{“}04 - 15 - 2021\text{”}$$

An ABAC policy model

- A Policy rule
  - decides on whether a subject s can access an object o in a particular environment e
  - is a Boolean function of the attributes of s, o, and e:

- Given all the attribute assignments of s, o, and e, if the function's evaluation is true, then the access to the resource is granted; otherwise the access is denied.

Example: ABAC for an online entertainment store

- The store streams movies to users for a flat monthly fee.

- The store must enforce the following access control policy based on the user's age and the movie's content rating

Movie Rating	Users Allowed Access
R	Age 17 and older
PG-13	Age 13 and older
G	Everyone

RBAC approach

- Roles
  - Adult, Juvenile, or Child
- Permissions
  - Can view R-rated movies, Can view PG-13-rated movies, and Can view G-rated movies.

- Both the user-to-role assignment and the permission-to-role assignment are manual administrative tasks.

ABAC approach

- The ABAC approach to this application does not need to explicitly define roles. Instead, whether a user u can access or view a movie m would be resolved by evaluating a policy rule such as the following:

$$
\begin{aligned}
&\text{R1:can\_access}(u,\ m,\ e) \leftarrow \\
&\quad (\text{Age}(u) \geq 17 \wedge \text{Rating}(m) \in \{\text{R},\ \text{PG-13},\ \text{G}\}) \vee \\
&\quad (\text{Age}(u) \geq 13 \wedge \text{Age}(u) < 17 \wedge \text{Rating}(\text{m}) \in \{\text{PG-13},\ \text{G}\}) \vee \\
&\quad (\text{Age}(u) < 13 \wedge \text{Rating}(m) \in \{\text{G}\})
\end{aligned}
$$

Compare RBAC & ABAC

- Suppose movies are classified as either New Release or Old Release, based on release date compared to the current date, and users are classified as Premium User and Regular User, based on the fee they pay. We would like to enforce a policy that only premium users can view new movies.

- With ABAC

```
R2:can_access(u, m, e) ←
   (MembershipType(u) = Premium)
   (MembershipType(u) = Regular ∧ MovieType(m) = OldRelease)
R3:can_access(u, m, e) ← R1 ∧ R2
```

Compare RBAC & ABAC

- Suppose movies are classified as either New Release or Old Release, based on release date compared to the current date, and users are classified as Premium User and Regular User, based on the fee they pay. We would like to enforce a policy that only premium users can view new movies.

- With RBAC

Standard RBAC Roles
Adult premium user role
Adult regular user role
Juvenile premium user role
Juvenile regular user role
Child premium user role
Child regular user role

Standard RBAC Permissions
Can view R rated new release
Can view R rated old release
Can view PG-13 rated new release
Can view PG-13 rated old release
Can view G rated new release
Can view G rated old release

Summary

- Overview of concepts
  - Access control principles
  - Subjects, objects, and access rights
- Access control models
  - Discretionary access control
  - Role-based access control
  - Attribute-based access control