CSCI262 : System Security

Week 3: Access Control 1

Schedule

- What is access control?
- Access control policies
- Role-based access control
- Attribute-based access control

What is access control?

- Access:
  - Able to do something, perhaps get somewhere.
  - Carry out an action
- Control:
  - To restrict or allow,
- **Access control**: being able to restrict or allow particular actions
- [SB18]: 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.

Access control context

- **Authentication:** Verification that the credentials of a user or other system entity are valid.
- **Authorisation:** The granting of a right or permission to a system entity to access a system resource.
- **Audit:** review and exam of system records and activities to ensure compliance with policies

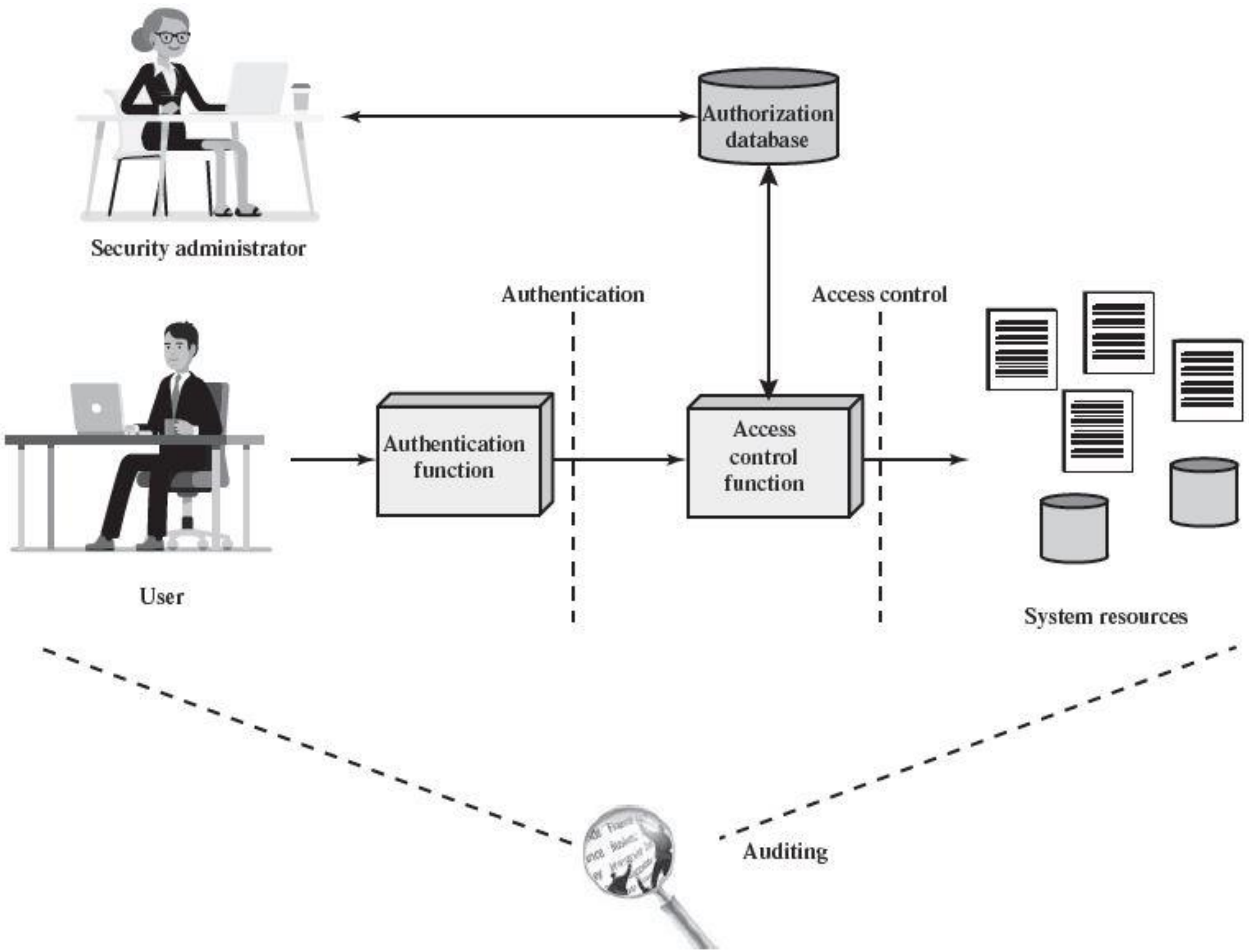


Figure 4.1 **Relationship Among Access Control and Other Security Functions**

Source: Based on [SAND94]

Question

- Do we always need both authentication and access control?
- Are there any circumstances where we might apply one but not the other?

Access control policies

- An access control policy dictates what types of access are permitted, under what circumstances, and by whom.
- Categories:
  - **Discretionary access control (DAC):** Users use their own discretion to specify who can access what
  - **Mandatory access control (MAC):** control access based on comparing security labels with security clearances
  - **Role-based access control (RBAC):** control access based on users' roles
  - **Attribute-based access control (ABAC):** control access based on users' attributes

Subjects, objects, access rights

- Subject: entities capable of accessing objects → active
    - Includes users and processes
- Object: entities (resources) that are used → passive
    - Includes files, directories, memory
- Access right: describe the way in which a subject may access an object.
    - read, write, execute, delete, create, search

Access control matrices

- The concept was developed independently by researchers in operating systems and databases.
- An **access control matrix** (ACM) model is defined in terms of state and state transitions.
- The **state** of a system is defined by a triplet (S, O, A):
    - S: A set of subjects.
    - O: A set of objects.
    - A: An access control matrix, A[S, O] with entries a(s,o).
        - a(s,o) lists the access rights of s on o.
        - Access rights specify the kind of access allowed for a subject on each object.

Example: Access Control Matrix

Objects Subjects	**File$_1$**	**File$_2$**	**Process$_1$**	**…**	**…**
Process$_1$	Read	Read Write			
Alice		Read	Execute		
Bob	Write		Execute		
Carol					
…					

Translate please

- Process$_1$ can read File$_1$:
  - If $A(\text{Process}_1, \text{File}_1) \supseteq$ Read
- Alice can execute Process$_1$:
  - If $A(\text{Alice}, \text{Process}_1) \supseteq$ Execute
- …

Advantages and Disadvantages of ACM?

- **Advantages:**
  - Allows for fast and easy determination of the access control rights for any subject-object pair
    - Just go to the cell of the matrix corresponding to this subject’s row and object’s column.
  - Gives administrators a simple, visual way of seeing the entire set of access control relationships all at once
  - The degree of control is as specific as the granularity of subject-object pairs.
- **Disadvantages:**
  - It is too big : n subjects and m objects result in a matrix with nm cells

Representations of Access Control

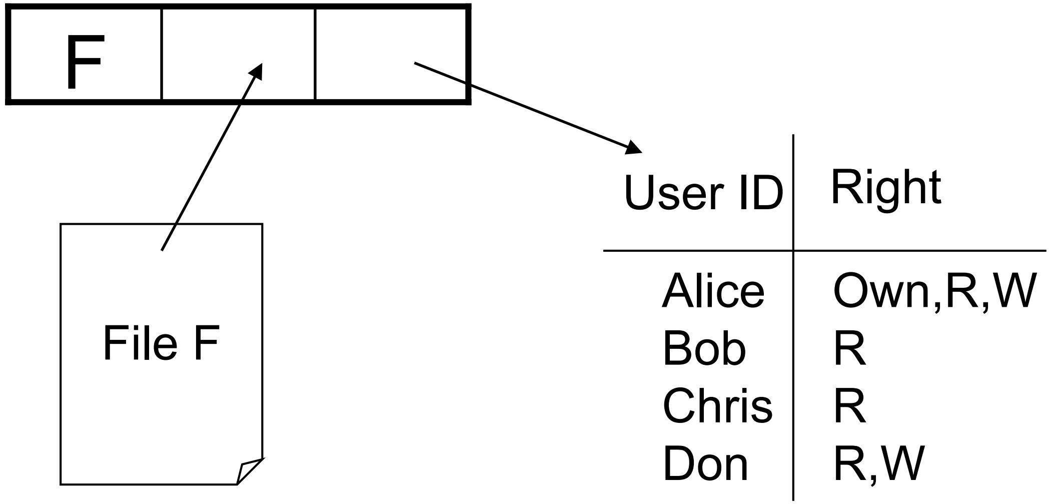
- **Access Control Lists (ACL):**
  - Contain access from the viewpoint of an object.
  - In other words, a column of the ACM.
  - For example:
    - File F : (A, Write), (B, Read).
- **Capabilities:**
  - These correspond to the viewpoint of a subject.
  - In other words, a row of the ACM.
  - For example:
    - ((Read, F1), (Write, F2)….)
  - Subject presents a capability to an object.

Access Control Lists

- A list of subjects that are authorised to access an object.
- Identity-based policies (such as individual-based and role-based policies) can be realised in a straightforward way.
- Maintenance of the list and enforcement of the access control are essentially the responsibility of the systems and environment surrounding the object.
- Access control list:

s1	s2	s3	...
r1	r2	r3	...

- What does it mean?
  - Subject s1, s2, s3 … have rights r1, r2, r3 … for this object.
  - The subjects could be individuals or roles (team leaders, managers…)



User ID	Right
Alice	Own,R,W
Bob	R
Chris	R
Don	R,W

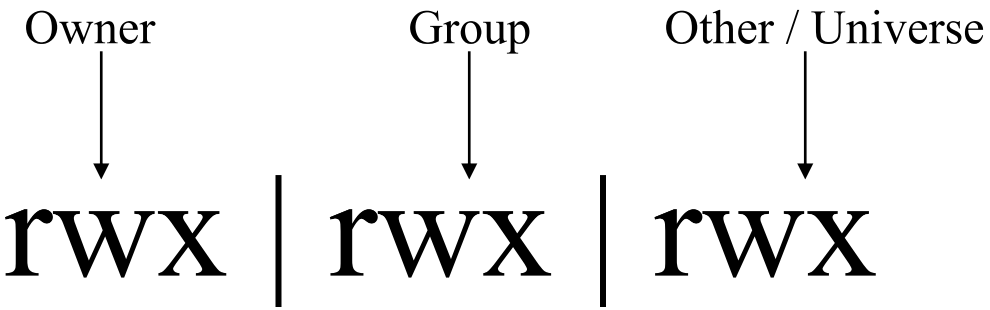
Access control lists are used to protect owned objects. The owner can confer/invoke rights by adding, deleting or modifying the entry for a user.

- In **Unix** a file has an access control list with three entries: owner, group, and other users.
- The type of access can be r, w, x (roughly).

```
$ ls -l | more
total 152
drwxr-xr-x   4 wsusilo  cs-uow       512 Aug 14  2007 111
drwxr-x---   5 greg     cs-uow       512 Nov  8  2004 112
drwxr-x---   2 david    csstf        512 Sep  8  2004 114.old
drwxr-x---   7 txia     other        512 Oct 18  2004 121
drwxr-x---   8 dfs      csstf        512 Dec  2  2003 121.2003
drwxr-x---   5 wsusilo  cs-uow       512 Sep  9  2003 121.old
drwxrwx---   4 koren    other        512 Apr  4  2006 124
lrwxrwxrwx   1 root     other         37 Jan  7 15:43 131 -> /web/itacs/documents/subjects/csci131
drwxr-x---   4 jrg      cs-uow       512 Jun  7  2001 1999.235
drwxrwxr-x   2 ian      csci203m    1024 May 30 11:46 203
drwxr-x---  10 greg     uowugc       512 Sep 20  1999 203.old
drwxr-sr-x   7 lukemc   csci204m     512 May 16 17:14 204
drwxr-x---   2 dfs      root         512 Jul 21  2007 204.dfs
drwxr-xr-x   7 lei      cs-uow       512 May 28 13:33 212
…
```

The permission string…

- The permission string - --- --- --- can be (if we ignore the first element) written in terms of permissions for the three classes:



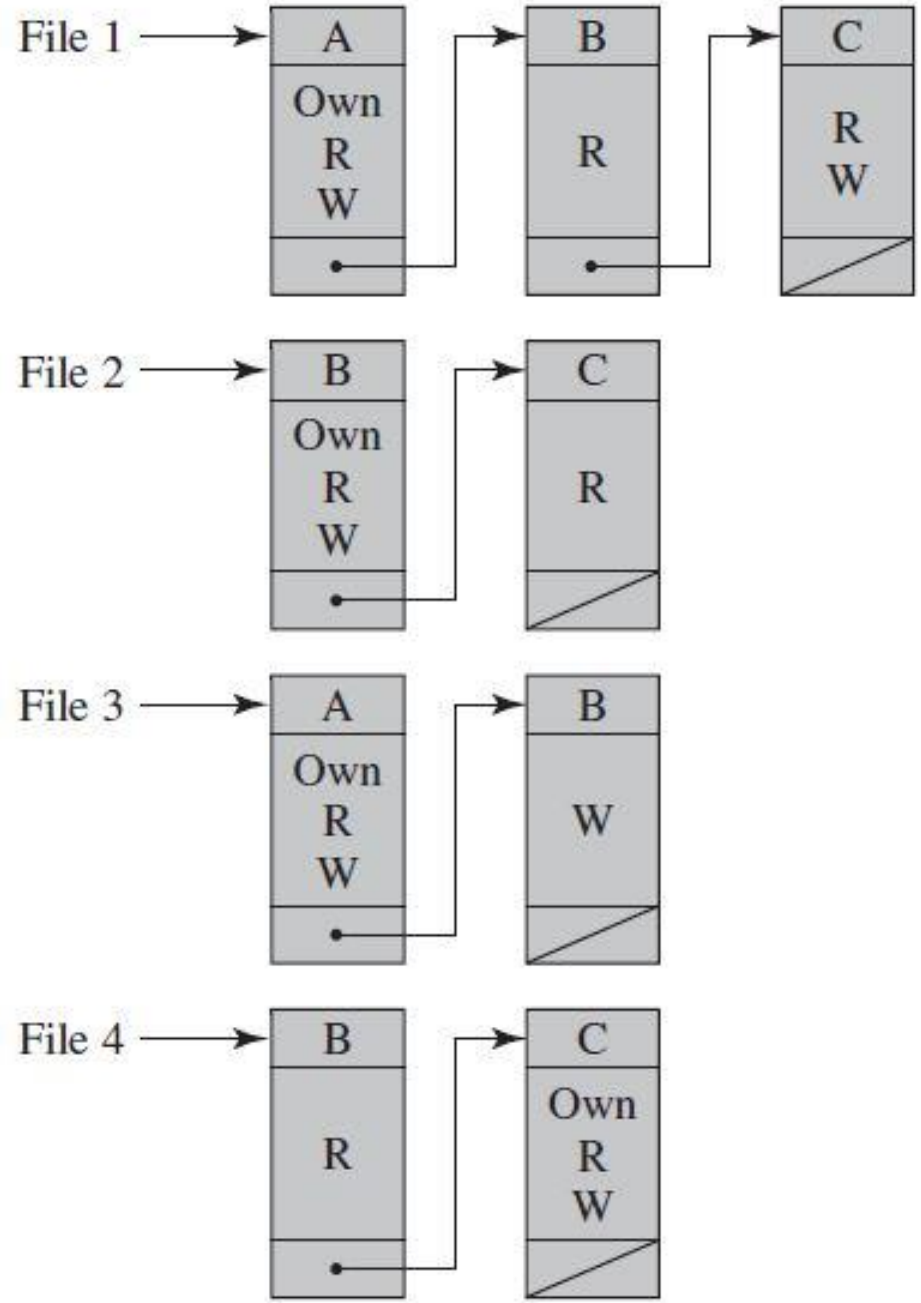
Owner

Group

Other / Universe

rwx | rwx | rwx

- The ‘R’ permission means read.
- The ‘W’ permission means write.
- The ‘X’ permission means execute on a file. In the context of a directory it means the directory can be searched.



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

Figure 4.2 in [SB18]

- File 1: (A,Own/R/W), (B,R), (C,R/W)
- File 2: (B,Own/R/W), (C,R)
- File 3: (A,Own/R/W), (B,W)
- File 4: (B,R), (C,Own/R/W)

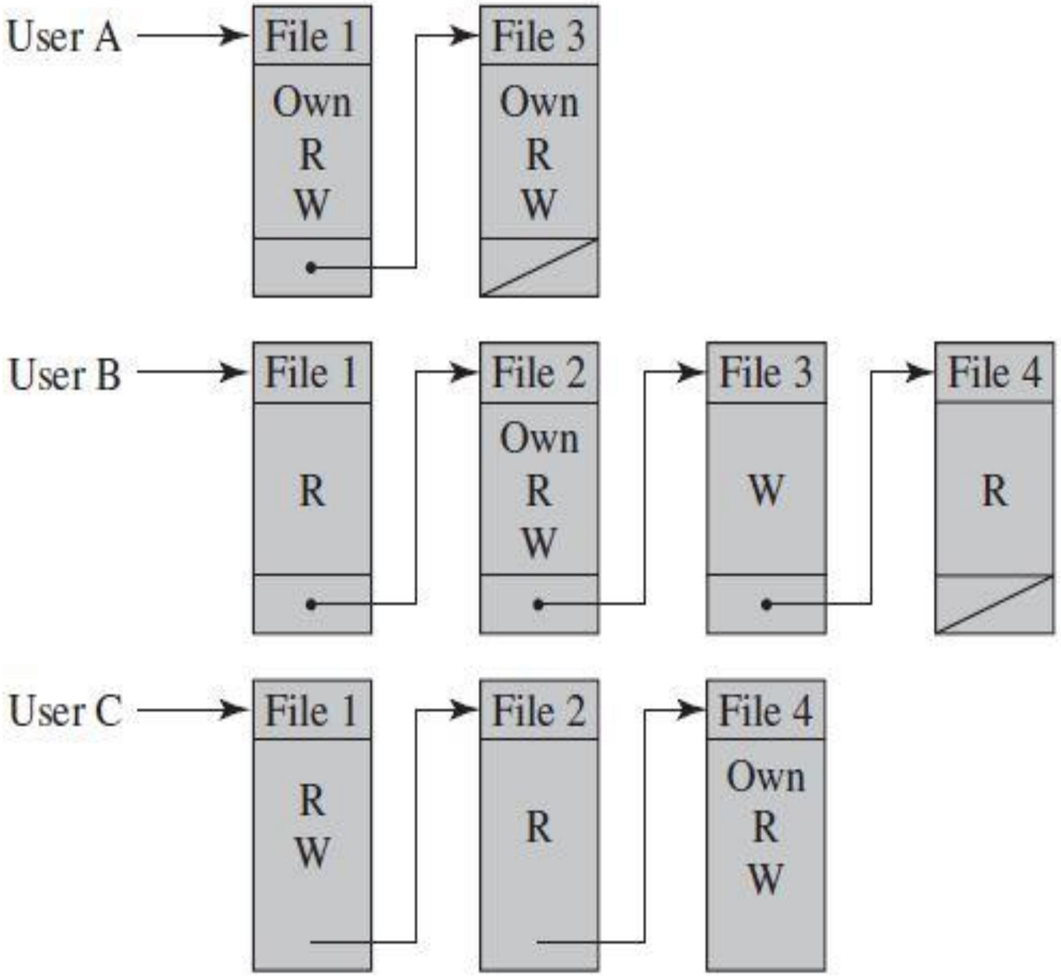
Advantages & disadvantages of ACL

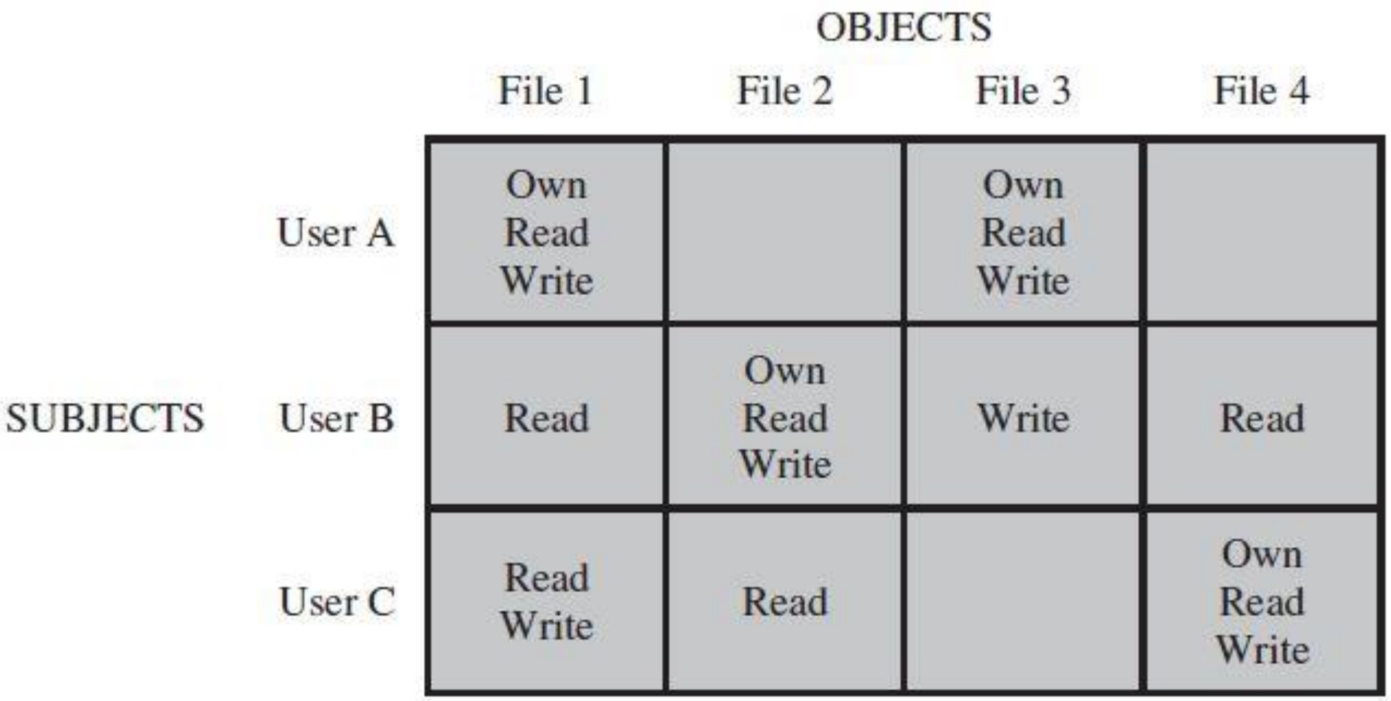
- **Advantages:**
- The main advantage of ACLs over access control matrices is size
- the ACL for an object can be stored directly with that object as part of its metadata, which is particularly useful for file systems.
  - The header blocks for files and directories can directly store the access control list of that file or directory
    - Thus, if the operating system is trying to decide if a user or process requesting access to a certain directory or file in fact has that access right, the system need only consult the ACL of that object.
- **Disadvantages:**
- Don’t provide an efficient way to enumerate all the access rights of a given subject.
  - In order to determine all the access rights for a given subject, s, a secure system based on ACLs would have to search the access control list of every object looking for records involving s.
  - Unfortunately, this computation is sometimes necessary

Capabilities

- Describes what a subject is capable of doing.
- Subject : With a list of (Object, Rights)
- Classical Capability System:
  - Capability : A triplet: <Object, Rights, Check>
  - Check = f(Object, Rights).
- We can think of a capability as a ticket that authorises the holder to access an object in a particular way.
  - The Check is there for authentication. It could be something like a message authentication code or digital signature (see CSCI361).
  - Capabilities are difficult to revoke.

- To obtain access an access request and the capability are transmitted to the appropriate server.
- Access Decision:
  - When the access request and capability arrive, the function f is applied to detect tampering.
  - If the capability passes => access is granted!





		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

Figure 4.2 in [SB18]

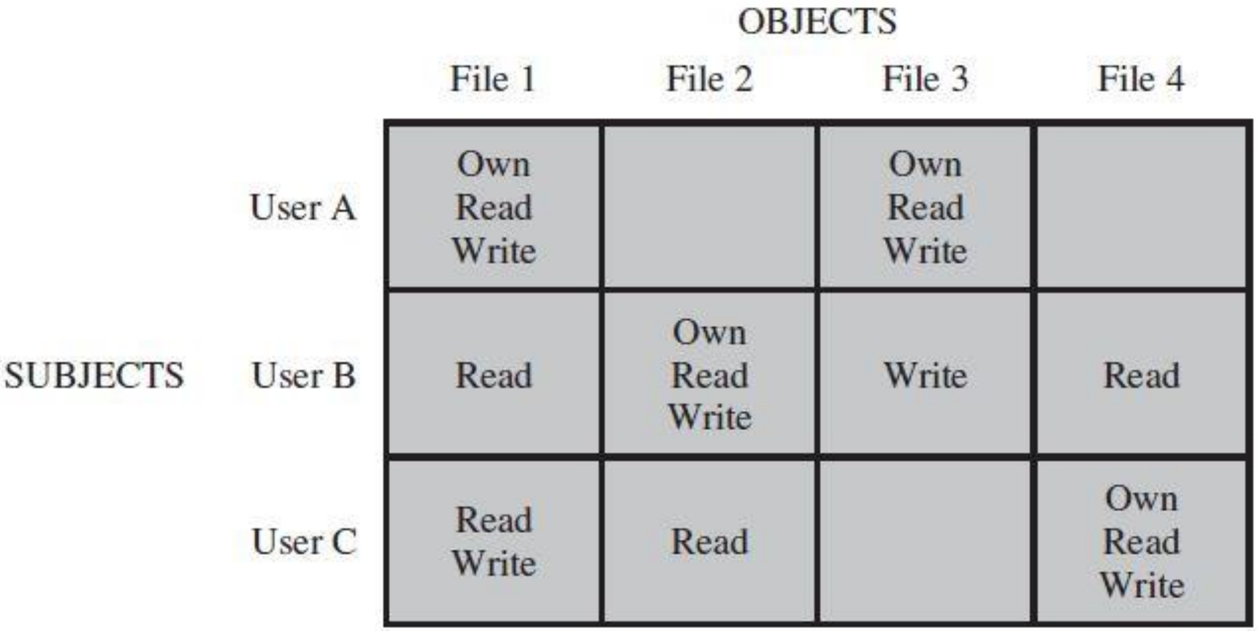
- User A: (File 1,Own/R/W), (File 3, Own/R/W)

Advantages & Disadvantages

- **Advantages**
  - Has the same advantage in space over the access control matrix as ACL
    - A system administrator only needs to create and maintain access control relationships for subject-object pairs that have nonempty access control rights
  - Makes it easy for an administrator to quickly determine for any subject all the access rights that that subject has.
    - Just read off the capabilities list for that subject
- **Disadvantages**
  - not associated directly with objects
    - Thus, the only way to determine all the access rights for an object o is to search all the capabilities lists for all the subjects

Sandhu & Samarati's authorization table

- The access control matrix may be sparse.
  - A more concise representation, but less common, is the authorization table.
- One row of the table is used for each allowed access triplet.
- The table can be sorted by Subject or Object to give equivalence to a capability list or an ACL, respectively.
- A relational database can easily implement an authorization table of this type. (More about relational database later)

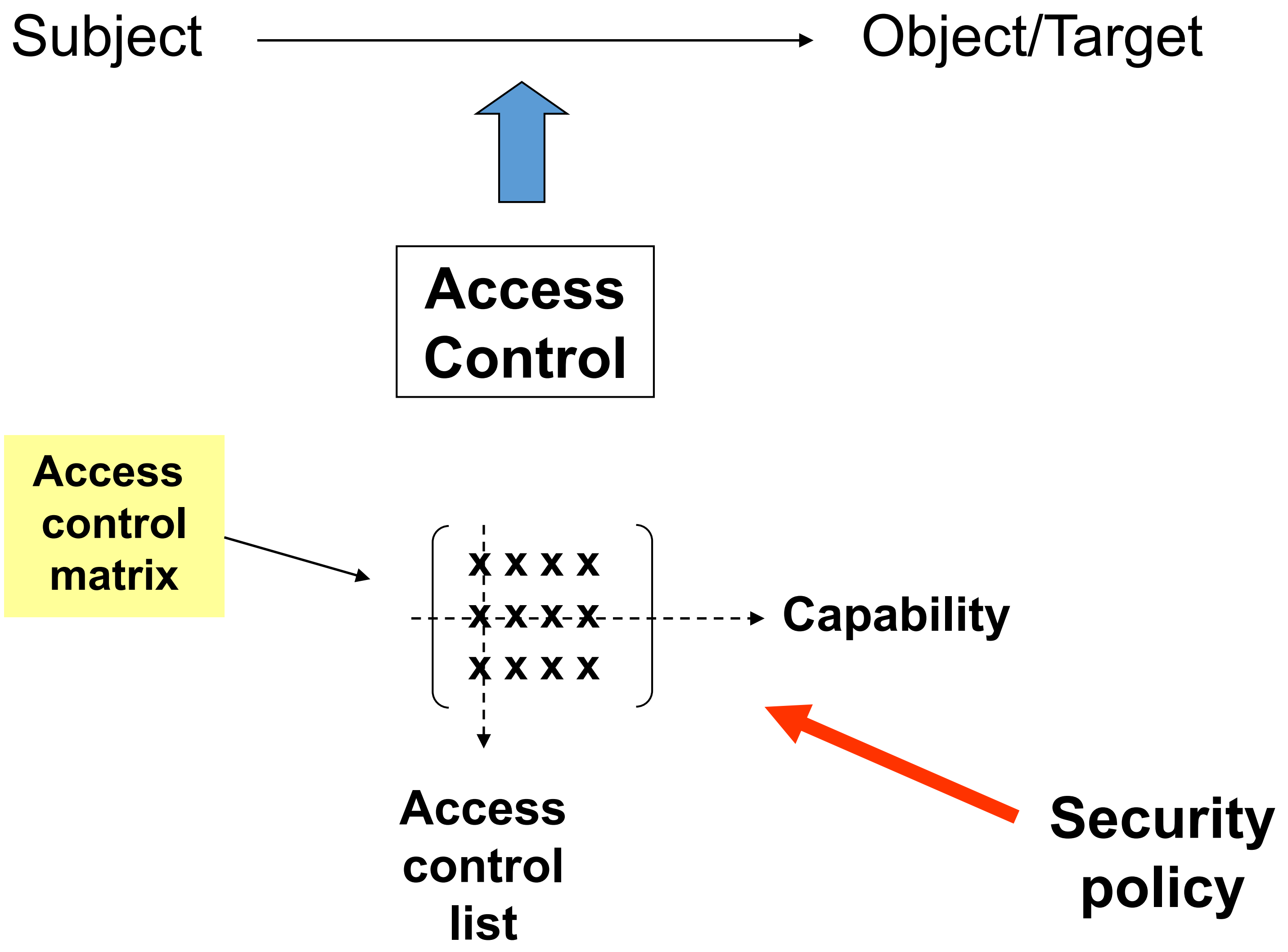


		OBJECTS			
		File 1	File 2	File 3	File 4
	User A	Own Read Write		Own Read Write	
SUBJECTS	User B	Read	Own Read Write	Write	Read
	User C	Read Write	Read		Own Read Write

Table 4.2 Authorization Table for Files in Figure 4.2

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 brief summary to now …



Subject

Object/Target

Access Control

Access control matrix

Capability

Access control list

Security policy

Intermediate Controls

- Group-based access control
- Previleges
- Role-based access control
- Protection ring

Group-based access control

- Users are assigned to groups.
    - Depending on the system policy, a user might be allowed to be a member of one group or multiple groups.
- Groups are given permissions to access objects.
- Each user has the permissions assigned to the group or groups it is a member of.
- We can think of grouping as being always on structures, relative to the roles we will look at soon.
    - For example, you are in a group whether you are logged in or not.

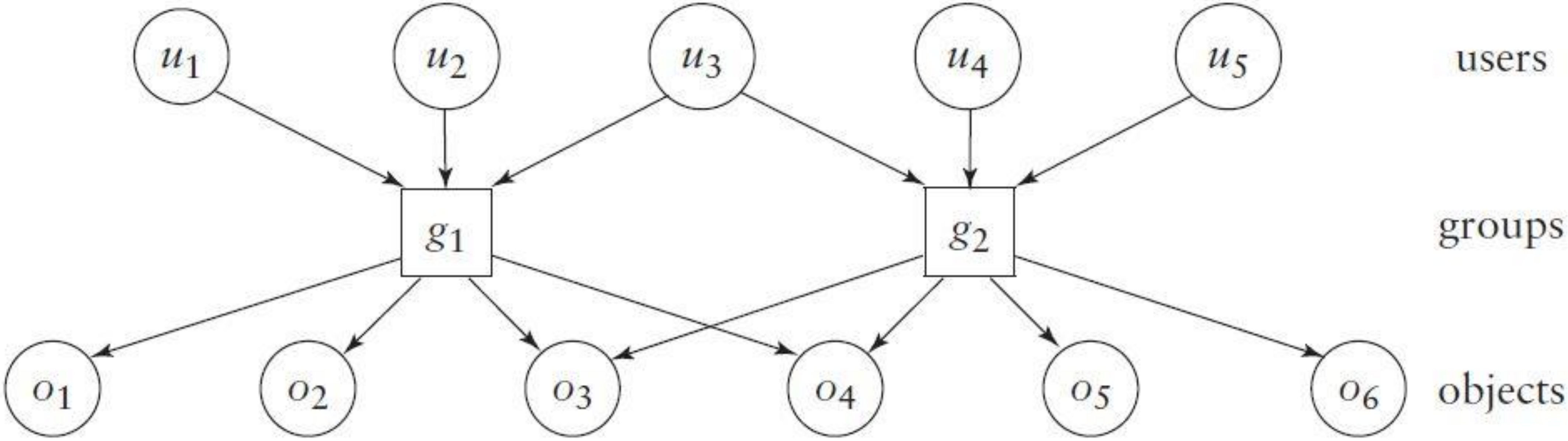


Figure 5.6 in [G11]: Groups serve as an intermediate access control layer

- All access permissions can be mediated through group membership
- For example: user u3 can access object o1 (being in group g1) and object o6 (being in group g2)

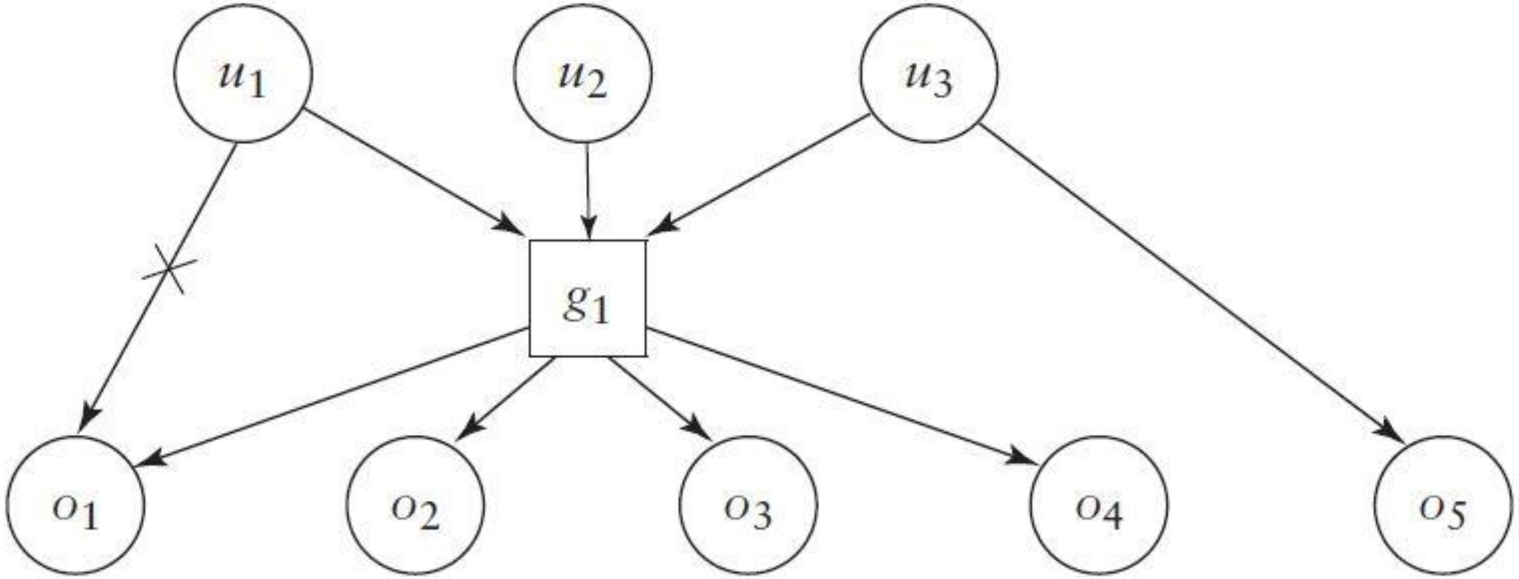
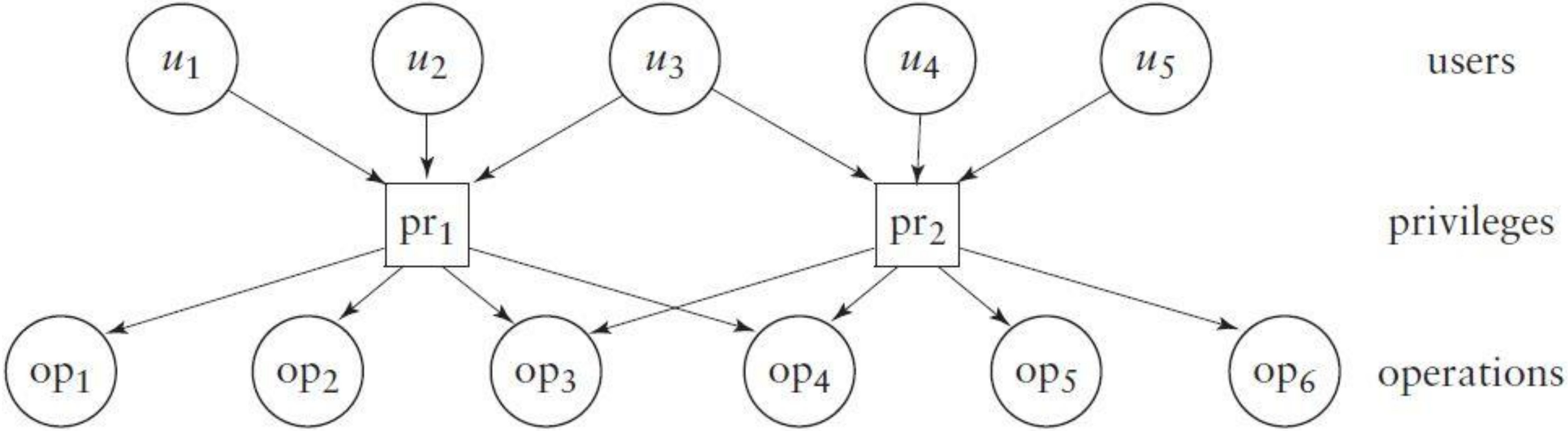


Figure 5.7 in [G11]: Access Control with Negative Permissions

- Notice the negative permission.
- A negative permission is an entry in an access control structure that specifies the access operations a user is not allowed to perform
- There is a conflict in the policies associated with s_1.
  - There would need to be a reference monitor policy to resolve this conflict.

Privileges : action grouping

- Privileges can be viewed as intermediate between subjects and actions or operations.
  - A subject is assigned privileges that allow the subject to execute certain operations, probably on certain objects.
  - Typically, privileges are associated with OS functions and relate to activities like system administration, or network access.



Role based access: An example

- In a banking environment there are several appropriate roles: Teller, Branch Manager, Customer, System Administrator, Auditor.
- A **Teller** has permission to modify a customer account with a deposit, carry out withdrawal transactions up to a specified limit, and query all account log entries.
- A **Branch Manager** has the same permissions as a teller but can also create and terminate accounts.

- A **Customer** is allowed to query the account log for his/her own account.
- A **System Administrator** can query all system log entries, activate/deactivate the system, but cannot read or modify customer account information.
- An **Auditor** can read any data in the system but modify nothing.

Role-based access cont

- Assign access rights to roles instead of individual users.
- Users are assigned to different roles.
    - A single user may be assigned multiple roles.
    - Multiple users may be assigned to a single role.

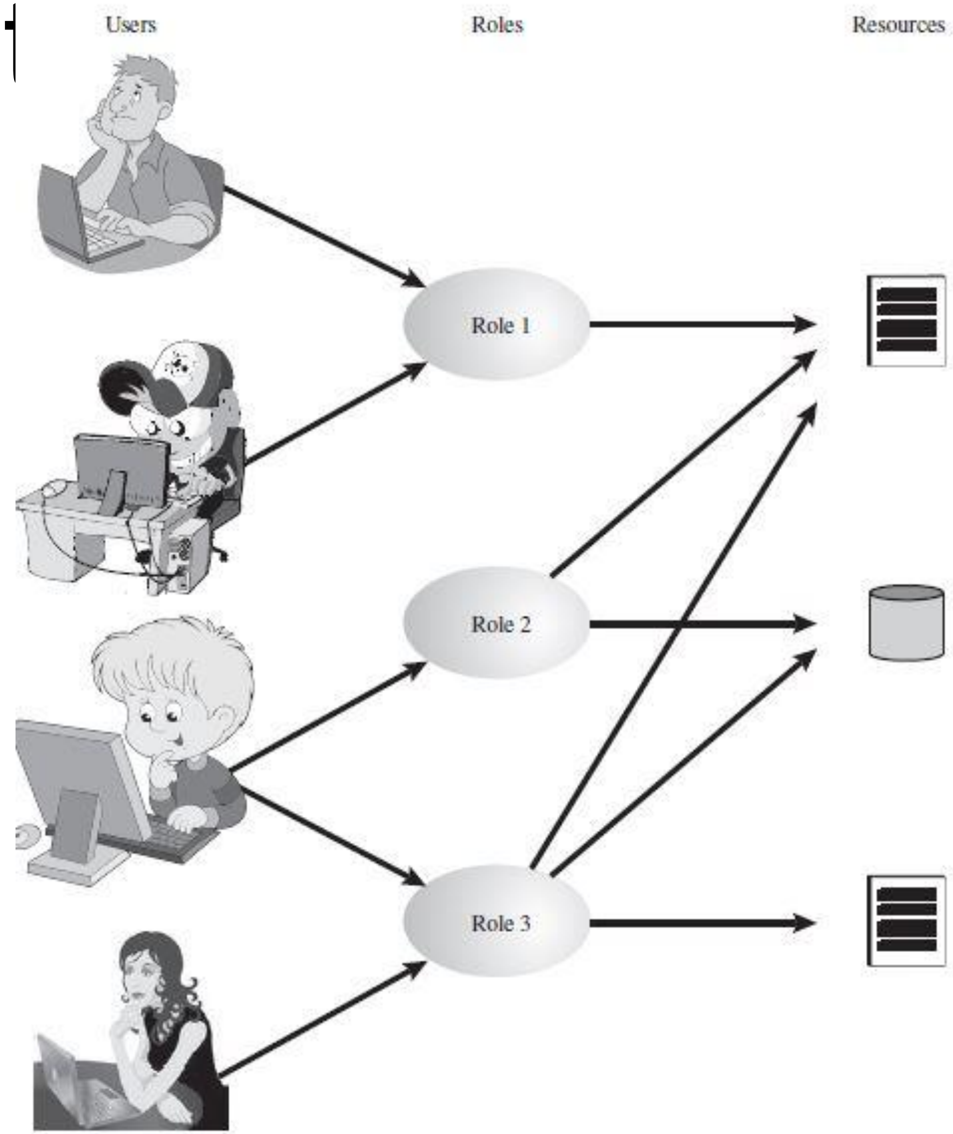
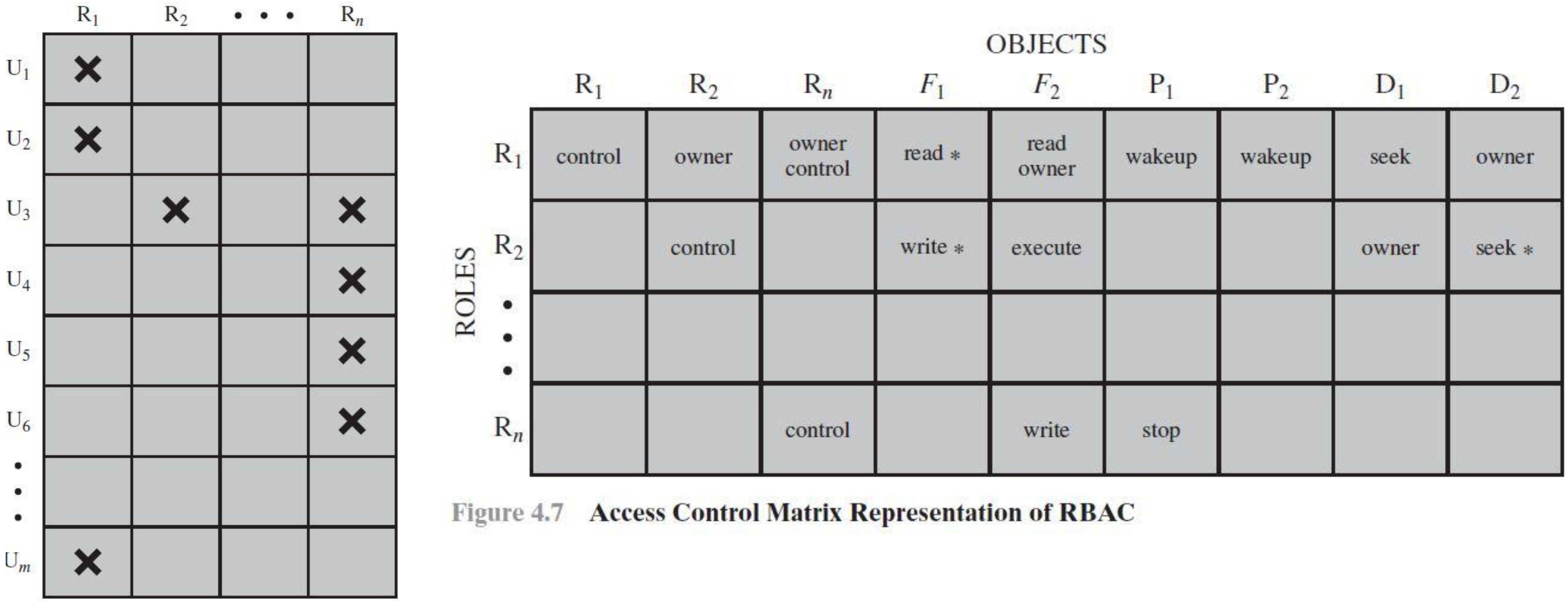


Figure 4.6 in [SB18]: Users, Roles, and Resources

ACM for RBAC



	R_1	R_2	• • •	R_n
U_1	✕			
U_2	✕			
U_3		✕		✕
U_4				✕
U_5				✕
U_6				✕
• • •				
U_m	✕			

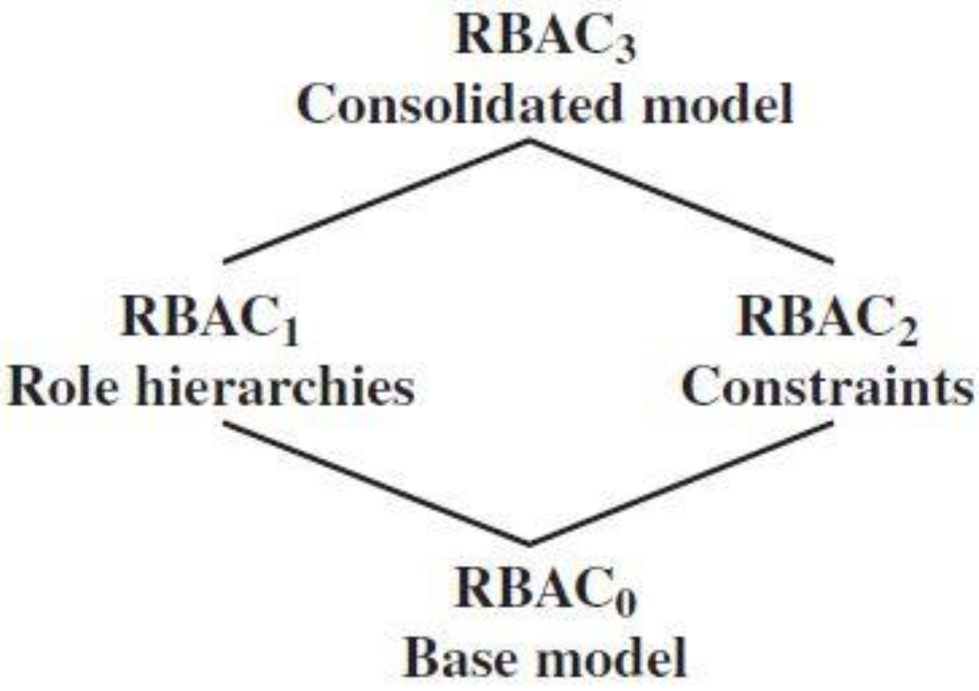
		OBJECTS								
		R_1	R_2	R_n	F_1	F_2	P_1	P_2	D_1	D_2
ROLES	R_1	control	owner	owner control	read *	read owner	wakeup	wakeup	seek	owner
	R_2		control		write *	execute			owner	seek *
	• • •									
	R_n			control		write	stop			

Figure 4.7 Access Control Matrix Representation of RBAC

- RBAC implements **principle of least privilege**:
    - Each role should contain minimum set of access rights needed for that role
    - A users is assigned to a role that enables him/her to perform only what is required for that role.
    - Multiple users assigned to the same role enjoy the same minimal set of access rights.

RBAC Models

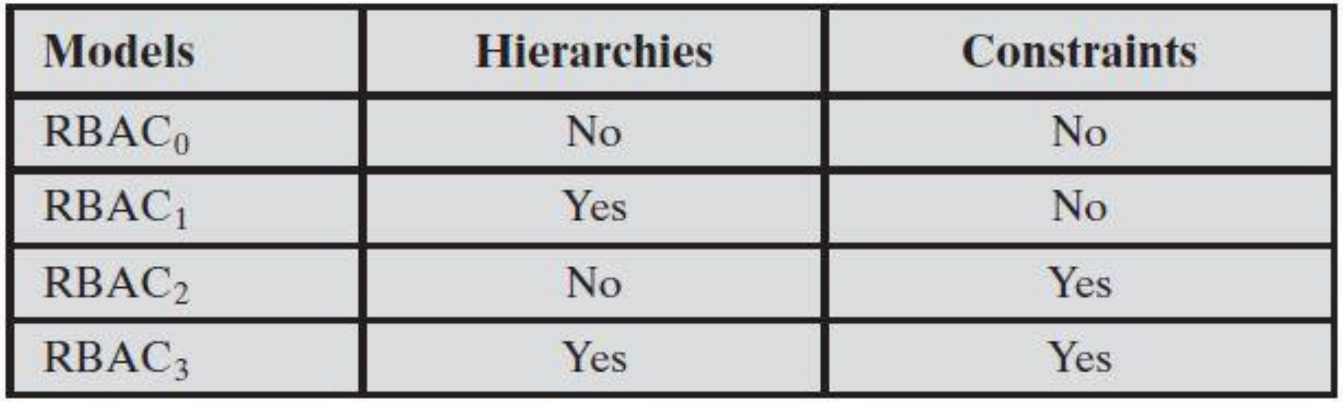
- There are four models that are related to each other.



(a) Relationship among RBAC models

Figure 4.8 in [SB18]

Table 4.4 **Scope RBAC Models**



Models	Hierarchies	Constraints
RBAC$_0$	No	No
RBAC$_1$	Yes	No
RBAC$_2$	No	Yes
RBAC$_3$	Yes	Yes

$RBAC_1$

- Role hierarchies:
  - Can reflect the hierarchical structure of roles in an organization.
  - Makes use of the concept of inheritance.

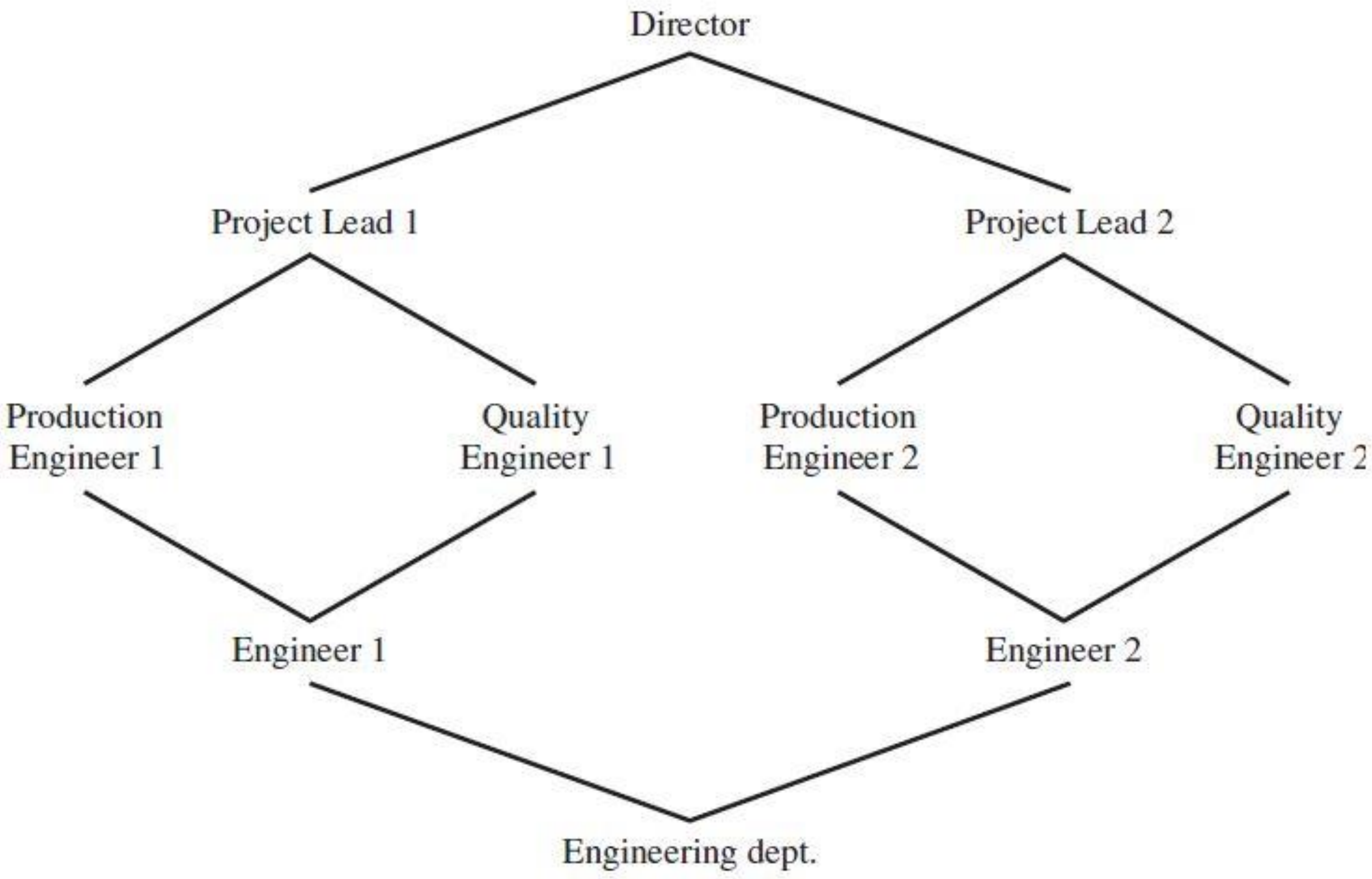


Figure 4.9 **Example of Role Hierarchy**

$RBAC_2$

- Constraints: mutually exclusive roles, cardinality, and prerequisite roles.
- **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:** set a maximum number with respect to roles.
  - Maximum number of users for a given role.
  - Maximum number of roles for a given user.
  - Maximum number of roles for a permission.
- **Prerequisite:** a user can only be assigned to a particular role if if it is already assigned to some other specified role.

Protection rings

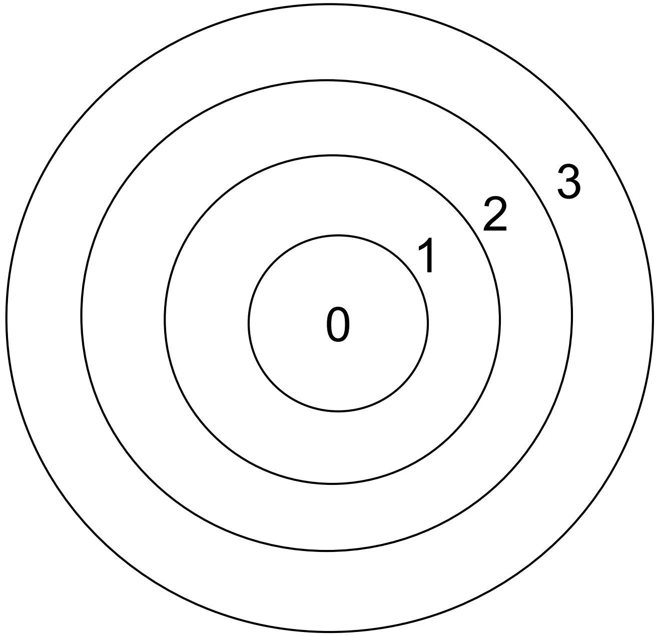
- Each subject or each object is assigned a number depending on its importance.
  - For an OS, it could be:

    0: OS kernel.

    1: OS.

    2: Utilities.

    3: User processes.
  - The numbers are compared to make decisions about access control.
  - Unix, Intel processors, etc. adopt this method.



Multilevel access control

- Protection rings are a special case of multilevel.
- The numbers are generalized into **security labels**.
- Subjects and objects have these security labels assignment to them, possibly using the same name but generally with different meanings.
- For subjects the labels correspond to clearances.
  - Think of every user having a clearance.
- For objects the labels correspond to classifications or sensitivity.
- The access relationship between security labels of the two types are governed by a series of rules.
- Multilevel/multilayer models are also referred to as *data flow models* → This is somewhat restrictive.

- A **Security label** is a set of information security attributes bound to an object or a subject.
    - The most common use is in supporting multilevel access control policies.
- When a subject makes an access request, a label is generated and attached to the request, by a trusted process.
    - Each object has a label bound to it, identifying it with a classification level.
- To process a request, a security server in the object environment compares the request label with the object label and applies policy rules, such as the Bell-LaPadula rules, to decide whether to grant or deny access.
    - We will talk about this later.

- Labels generated in one security domain may or may not be significant in another domain:
  - Consider, for example, labels of identical format in two different organizations or in parts of a single organization.
  - Information classified as confidential in the first organization, or part thereof, should generally not be disclosed to the persons with confidential clearance in the other organization, or other part thereof.
  - Access control models taking into account “horizontal” structures are referred to as multi-lateral, more on this later.

Multilevel policies: An example

- We need to describe whether a particular attempted action will be allowed as a function of the relevant labels, using policies.
- Example:
    - Labels: Our sensitivity levels are

      Top secret
      Secret
      Confidential
      Unclassified
    - Policy: An object can only be accessed by a subject with a clearance level as high as the object classification.
        - This is somewhat vague … we will look at BLP very soon and this will be clarified.

Attribute-based Access Control (ABAC)

- This is a relatively new approach to access control.
- In this approach we can construct more complex authorisation statements based on attributes associated with subjects, objects, operations, and the environment.
  - It doesn’t have the subject focus on RBAC.
- The flexibility is it’s upside, the performance cost, checks per access, it’s downside.

ABAC elements

- **Attributes**: Defined for entities.
- **Policy Model**: Defines the policies – so the rules and relationships for governing what's allowed and what's disallowed.
- **Architecture**: This is the infrastructure used to manage requests, at the policy and enforcement levels, and carry out the interactions with the sources of attributes.

Type of ABAC Attributes

- Attributes are characteristics defining specific aspects.
- Subject attributes:
  - Define identity and characteristics of the subject.
  - Examples: Name, age, job title, roles, …
- Object attributes:
  - As above but for an object.
  - Examples: File name, file title, creation date, …
- Environment attributes:
  - Largely only lightly used.
  - Examples: Date, time, attacker activity,…

ABAC Logical Architecture

1. A subject requests to an object
2. The access control mechanism is governed by a set of rules (2a) that are defined by a preconfigured access control policy to determine authorisation.
3. Allow/deny access.

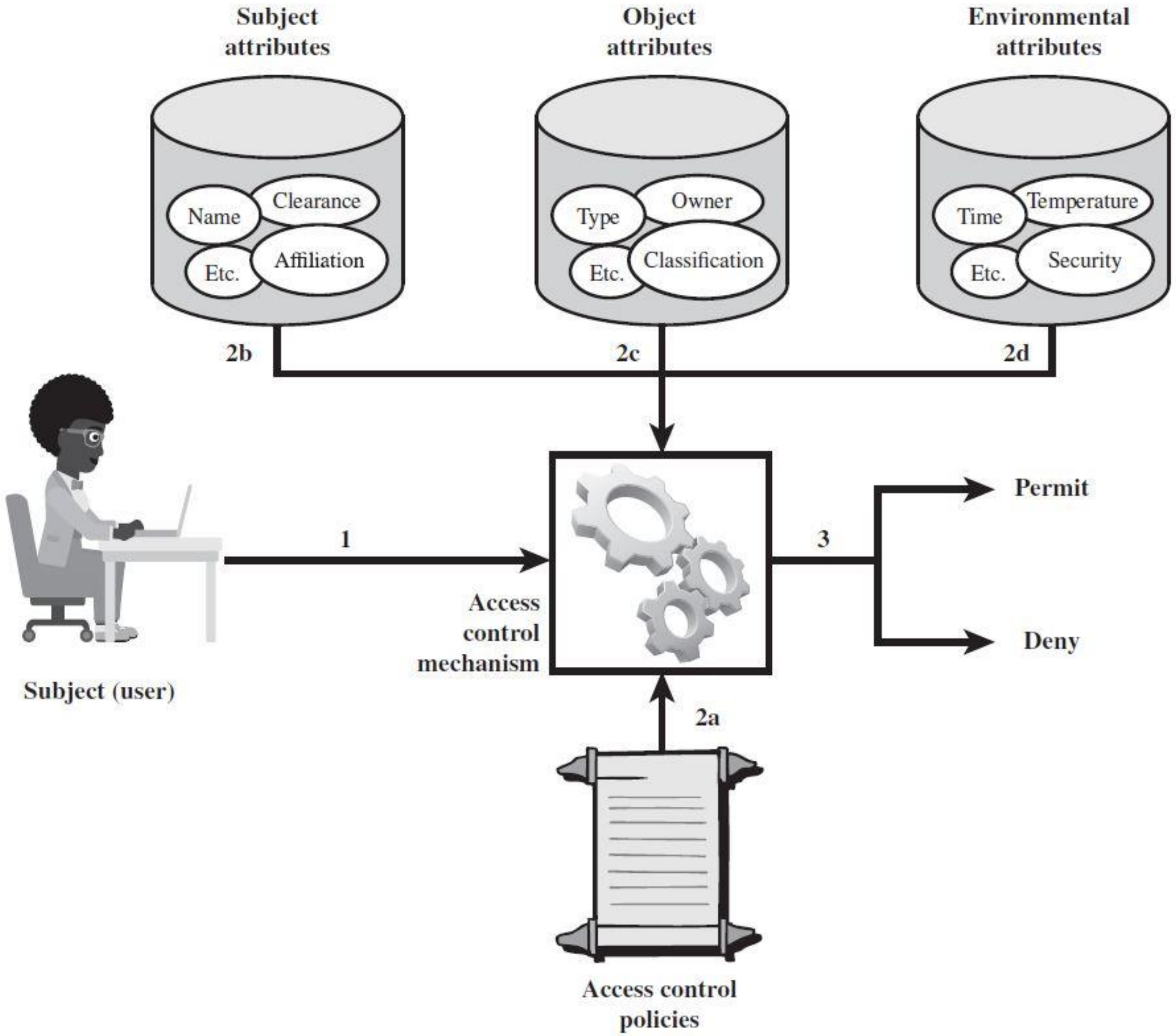


Figure 4.10 in [SB18]