Chapter 5

Access Control

Table of Contents

Access Control Principles 3
Subjects, Objects and Access Rights 9
Access Control Matrix and Capability Lists 13
Discretionary Access Control 17
Role Based Access Control...19
Attribute-Based Access Control.......................................26
Identity, Credentials, and Access management...............29
Trust Frameworks 33

Access Control

Access Control is a method of limiting access to a system or resources.

Access control refers to the process of determining who has access to what resources within a network and under what conditions.

Its main goal is to ensure that only authorized users can access specific resources, thereby protecting the confidentiality, integrity, and availability of information.

It is a fundamental concept in security that reduces risk to the business or organization.

Basically, access control is of 2 types:

- **Physical Access Control:** Physical access control restricts entry to campuses, buildings, rooms and physical IT assets.
- **Logical Access Control:** Logical access control limits connections to computer networks, system files and data.

Components of Access Control

1. **Authentication**: The process of verifying the identity of a user or system. Common methods include passwords, biometrics, and multi-factor authentication (MFA).
2. **Authorization**: Determines what an authenticated user is allowed to do. This includes defining permissions and privileges to access specific resources or perform certain actions.
3. **Accounting (or Auditing)**: Tracking user activities and access patterns to ensure compliance with security policies and to detect and respond to security incidents.

Access Control Principles

Access control principles are fundamental guidelines that help ensure the effective implementation and management of access control in information security.

These principles are designed to protect resources, maintain data integrity, and ensure the appropriate level of confidentiality.

Here are the key access control principles:

1. Principle of Least Privilege (PoLP)

This principle states that users should be granted the minimum level of access – or permissions – necessary to perform their job functions. By limiting access, the potential for misuse or accidental modification of sensitive data is reduced.

- **Example**: An employee in the marketing department should not have access to financial records or server configuration settings.

2. Separation of Duties (SoD)

Separation of duties involves dividing tasks and privileges among multiple users to reduce the risk of fraud and errors. This principle ensures that no single individual has control over all aspects of any critical function or system.

- **Example**: In a financial system, one person might be responsible for creating vendor payments, while another person is responsible for approving those payments.

3. Role-Based Access Control (RBAC)

RBAC assigns access rights based on user roles within an organization. Each role is associated with specific permissions, making it easier to manage and enforce access controls.

- **Example**: A role of "HR Manager" might have access to employee records, while a role of "HR Assistant" might only have access to less sensitive information.

4. Need to Know

Access to information or resources is restricted to individuals who need it to perform their job duties. This principle is often used in conjunction with the principle of least privilege.

- **Example**: Only employees working on a specific project can access the project's files and data.

5. Mandatory Access Control (MAC)

Under MAC, access rights are regulated by a central authority based on multiple levels of security. Users cannot alter access policies, and access is granted based on predefined policies and classifications.

- **Example**: Classified information in a government organization might be labeled as "Top Secret," and only individuals with the appropriate clearance level can access it.

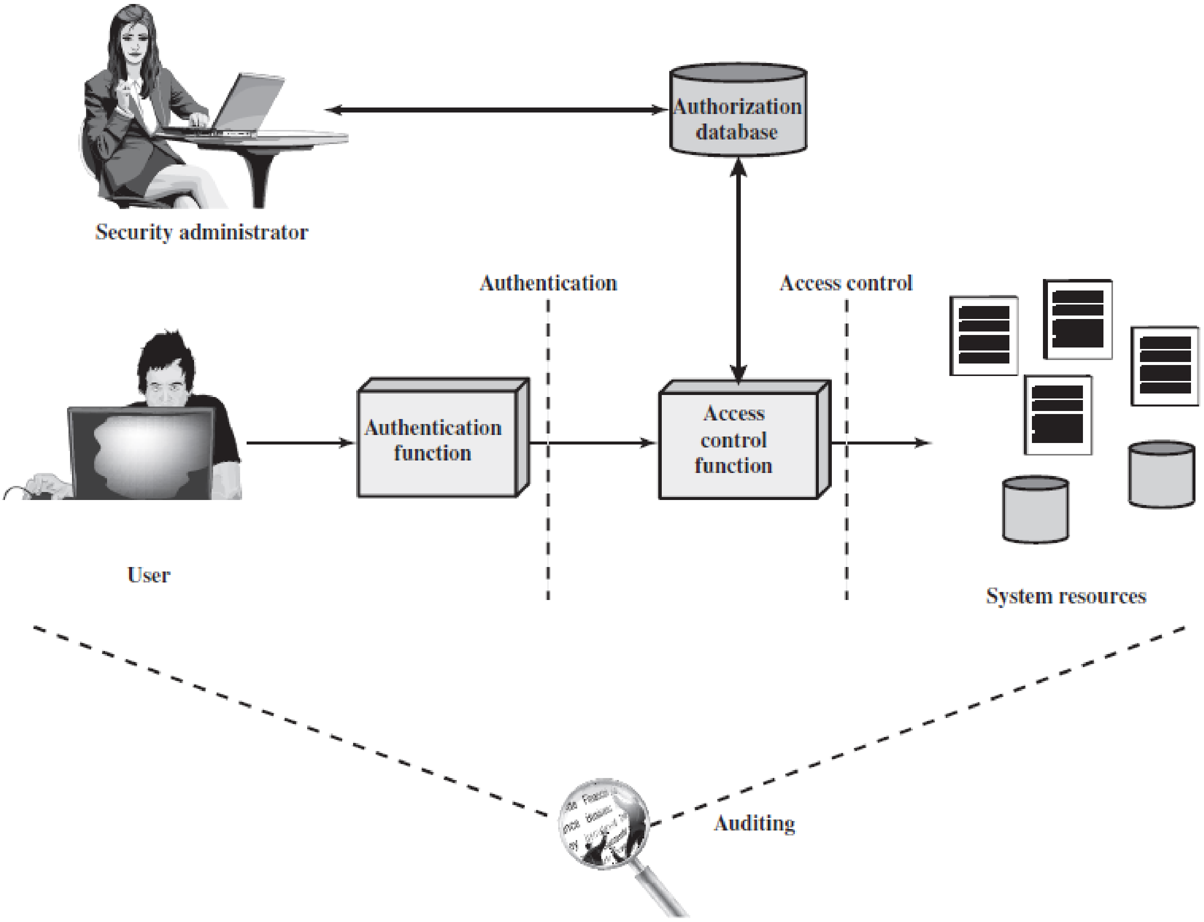


Figure: Relationship Among Access Control and Other Security Functions

Access control policies

Access control policies are rules that determine who can access certain resources and under what conditions.

Discretionary Access Control (DAC)

- **What it is**: Access is controlled based on the identity of the person trying to access the resource.
- **How it works**: You can give or take away access to resources as you see fit.
- **Example**: You own a document and you decide who can read or edit it. If you want your friend to edit the document, you give them permission.

Mandatory Access Control (MAC)

- **What it is**: Access is controlled based on strict rules about the sensitivity of the resource and the clearance level of the user.
- **How it works**: You cannot change who can access the resource based on your own decisions. The system enforces strict rules.

- **Example**: In a government system, a document labeled "Top Secret" can only be accessed by someone with "Top Secret" clearance. Even if you have access, you can't just let anyone else see it.

Role-Based Access Control (RBAC)

- **What it is**: Access is controlled based on the roles that users have within an organization.
- **How it works**: Users are assigned roles (like manager, employee, etc.), and each role has predefined access permissions.
- **Example**: In a company, a manager can approve expenses and access employee records, but an employee can only submit expenses and view their own records.

Attribute-Based Access Control (ABAC)

- **What it is**: Access is controlled based on various attributes of the user, the resource, and the environment.
- **How it works**: The system evaluates multiple factors (attributes) to decide if access should be granted.

- **Example**: A user can access a financial report only if they are in the finance department (user attribute), the document is marked as non-confidential (resource attribute), and they are accessing it during business hours (environmental condition).

Subjects, Objects and Access Rights

Subjects

A **subject** is an entity capable of accessing objects. Typically, a subject is a process that represents a user or an application. The process inherits the attributes and access rights of the user or application it represents. Subjects are accountable for their actions, and audit trails can record their activities to ensure accountability.

Classes of Subjects:

1. **Owner**:
    - The creator of a resource or file.

  - Could also be a system administrator for system resources.
  - For project resources, ownership might be assigned to a project administrator or leader.

2. **Group**:
   - A named group of users granted access rights to certain resources.
   - Membership in the group is enough to exercise these access rights.
   - Users can belong to multiple groups.
3. **World**:
   - Users who have the least amount of access.
   - They can access the system but are not included in the owner or group categories for a resource.

Objects

An **object** is a resource to which access is controlled. Objects contain or receive information and can be various types of data structures or resources within a system.

Examples include:

- Records
- Blocks
- Pages
- Segments
- Files
- Portions of files
- Directories
- Directory trees
- Mailboxes
- Messages
- Programs

Access Rights

Access rights describe the ways in which a subject may interact with an object. These rights determine what actions a subject can perform on an object.

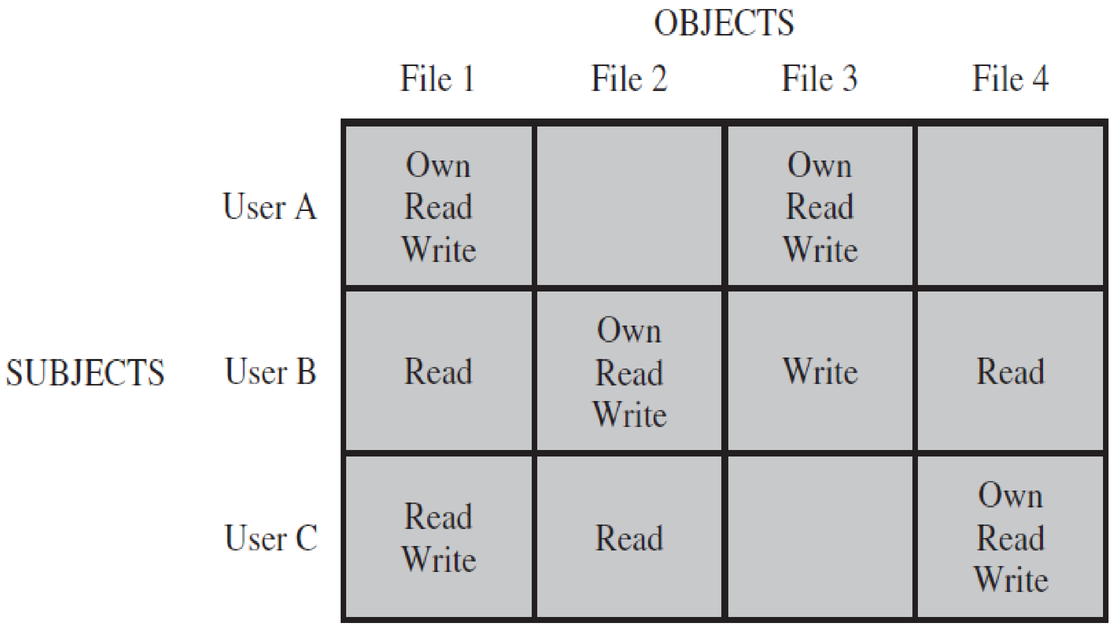
Common access rights include:

1. **Read**:
   - The subject may view information in a system resource.
   - This includes the ability to copy or print information.
2. **Write**:
   - The subject may add, modify, or delete data within a system resource.
   - Write access includes read access.
3. **Execute**: The subject may run specified programs.
4. **Delete**: The subject may remove certain system resources, such as files or records.
5. **Create**: The subject may create new files, records, or fields.
6. **Search**: The subject may list files in a directory or perform search operations within a directory.

Access Control Matrix and Capability Lists

Access Control Matrix

- It is a table that defines access permissions between specified subjects and objects
- Rows of ACM corresponds to users/subjects/groups
- Columns corresponds to resources that need to be protected
- ACM [U,O] defines what access rights user U has for object O
- It is also called Lampson's access control 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 Control List:

Access Control lists can be created by splitting the access matrix column-wise.

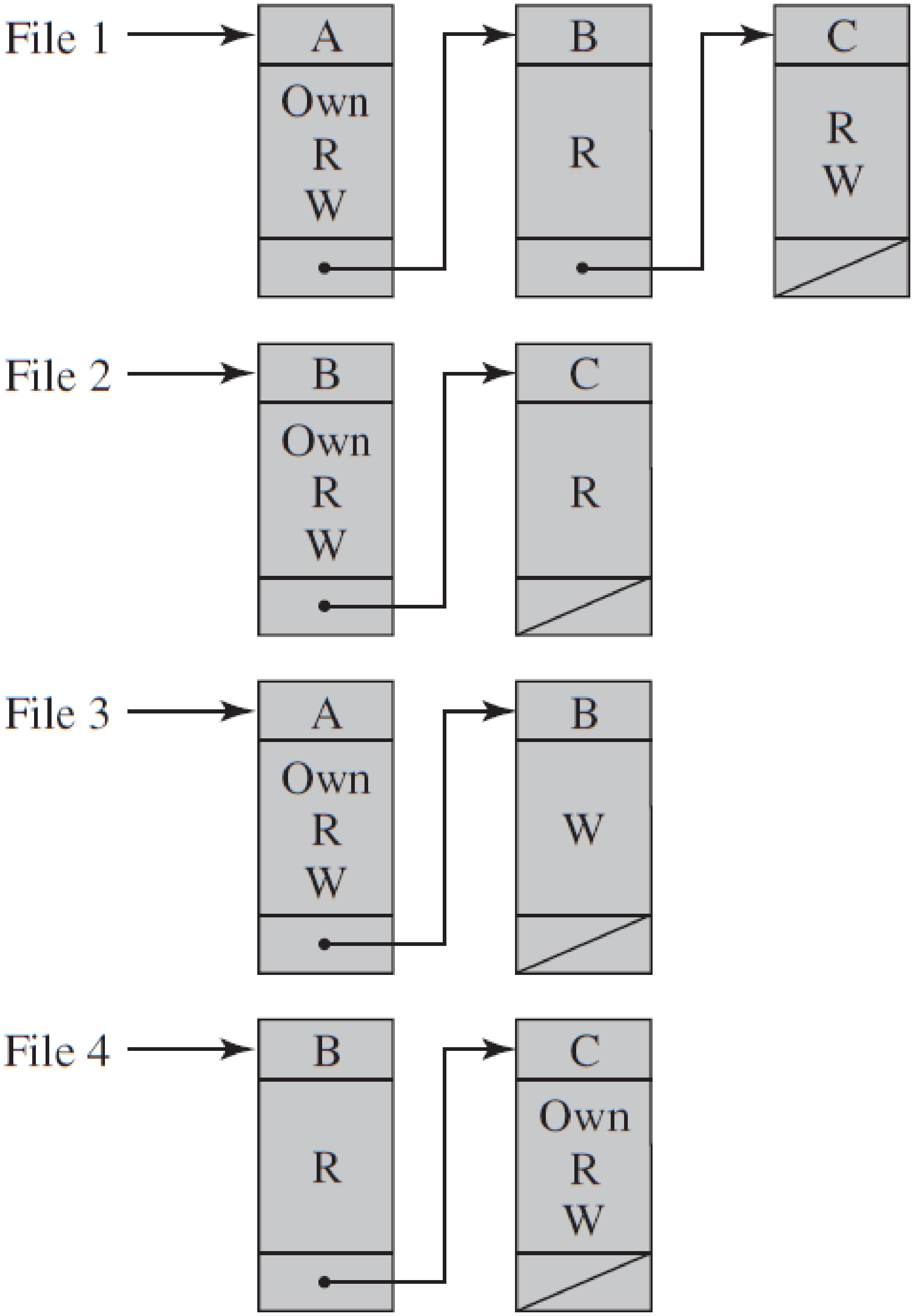
Access Control List is the object-wise list that specifies the list of subjects that have access to a particular object along with their access right.

Capability Lists:

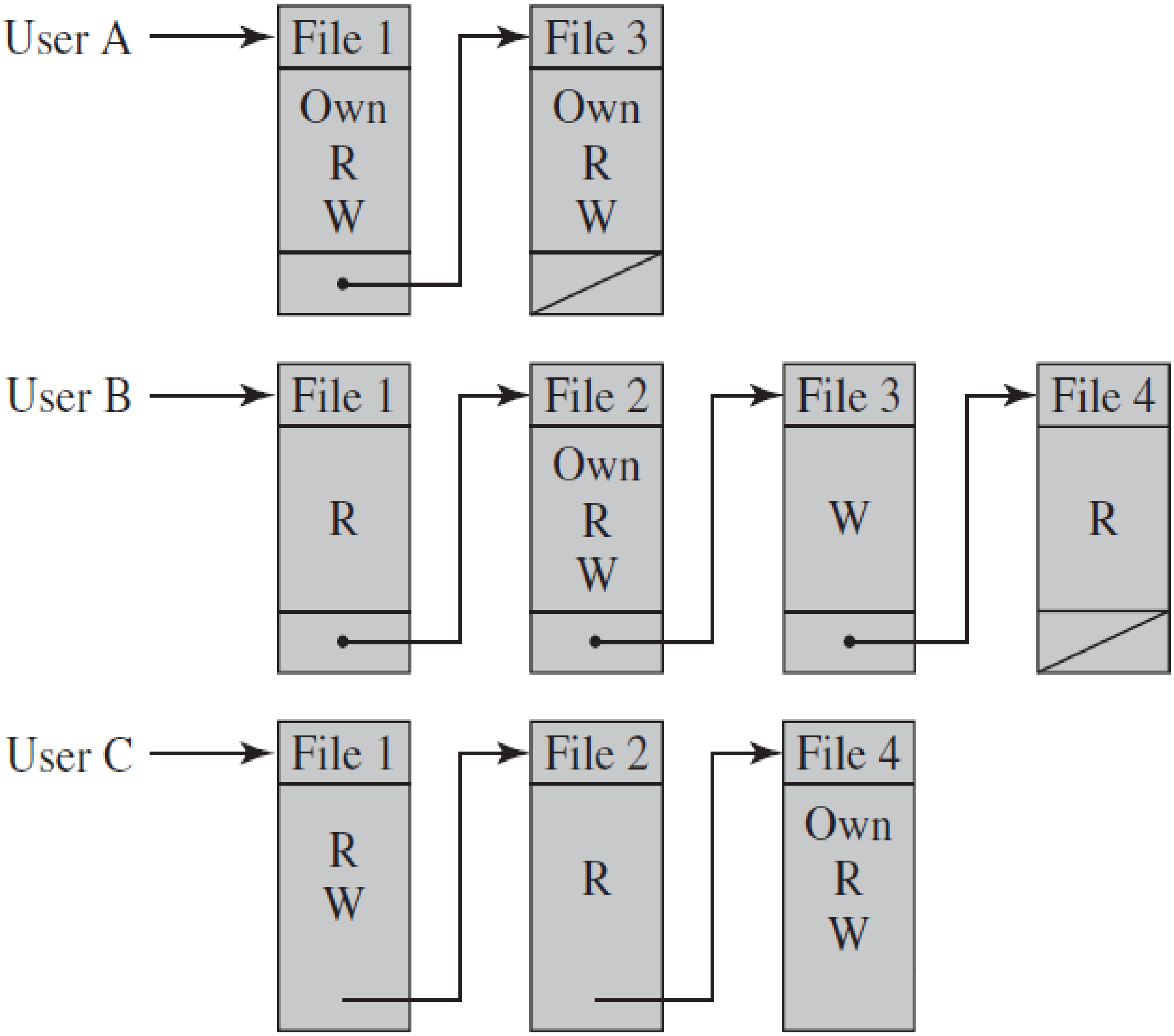
Capability lists can be created by splitting the access matrix row-wise.

A capability list is a subject-wise list that specifies the list of rights the subject has for every object.

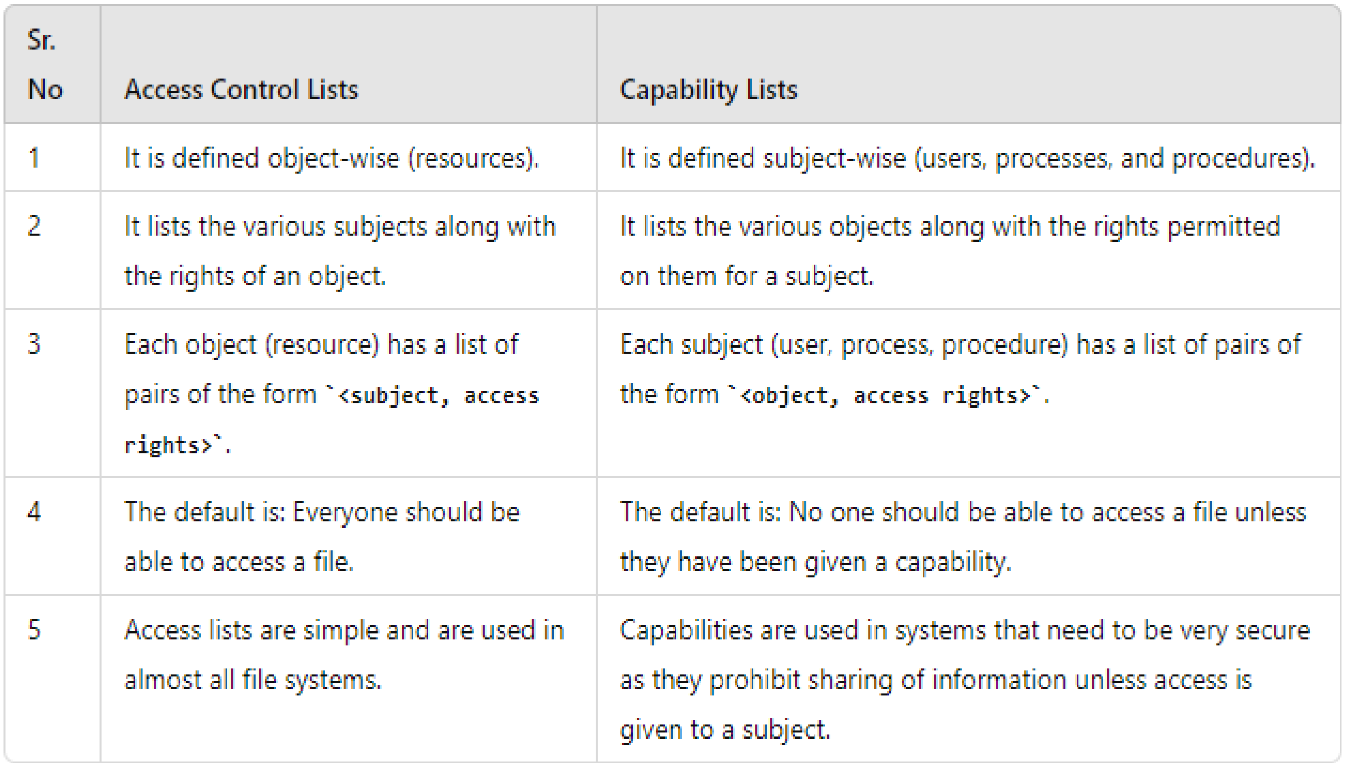
Thus, the capability list of a user or a process or domain is a list of rights that it has on the various objects.



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



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



Sr. No	Access Control Lists	Capability Lists
1	It is defined object-wise (resources).	It is defined subject-wise (users, processes, and procedures).
2	It lists the various subjects along with the rights of an object.	It lists the various objects along with the rights permitted on them for a subject.
3	Each object (resource) has a list of pairs of the form `` `<subject, access rights>` ``.	Each subject (user, process, procedure) has a list of pairs of the form `` `<object, access rights>` ``.
4	The default is: Everyone should be able to access a file.	The default is: No one should be able to access a file unless they have been given a capability.
5	Access lists are simple and are used in almost all file systems.	Capabilities are used in systems that need to be very secure as they prohibit sharing of information unless access is given to a subject.

Discretionary Access Control

DAC is identity-based access control. DAC mechanisms will be controlled by user identification such as username and password. DAC is discretionary because the owners can transfer objects or any authenticated information to other users. In simple words, the owner can determine the access privileges.

Attributes of DAC –

1. Users can transfer their object ownership to another user.

2. The access type of other users can be determined by the user.
3. Authorization failure can restrict the user access after several failed attempts.
4. Unauthorized users will be blind to object characteristics called file size, directory path, and file name.

Example of DAC

Consider a typical file system where users create files and directories. Here's how DAC works in this scenario:

1. **File Creation**:
   - When a user (e.g., Alice) creates a file, she becomes the owner of that file.
   - Alice can set permissions to specify who else can access her file and what they can do (read, write, execute).
2. **Setting Permissions**:
   - Alice can give read access to Bob and write access to Carol.

    - This means Bob can only view the contents of the file, while Carol can modify it.

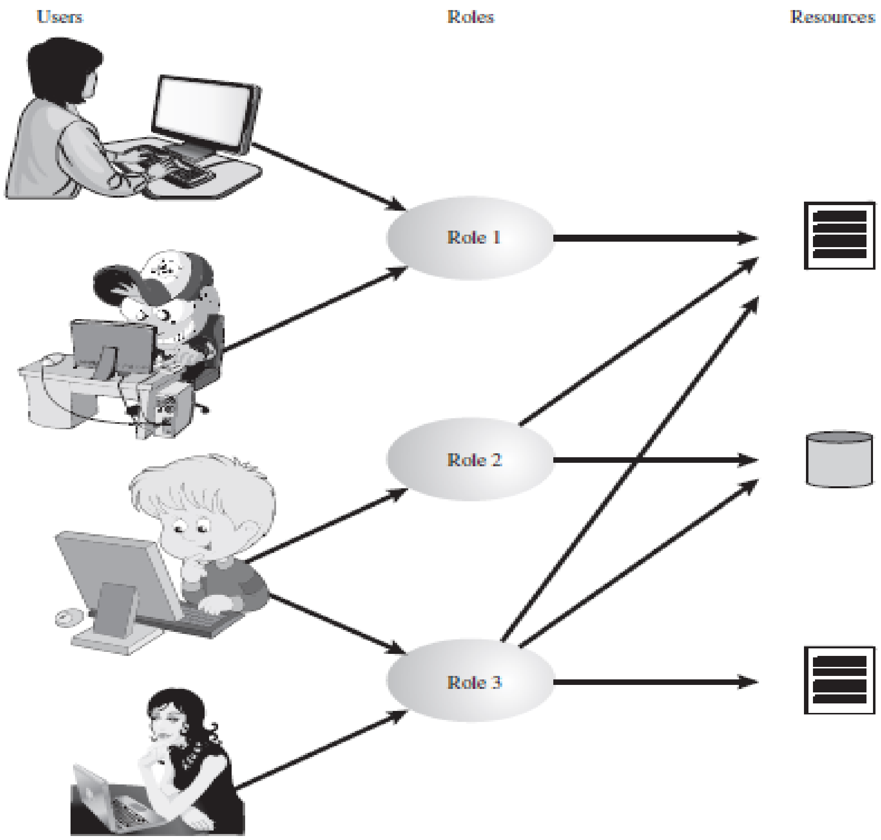
3. **Changing Permissions**:
    - At any time, Alice can change the permissions, for example, revoke Bob’s read access or give execute permission to Dave.

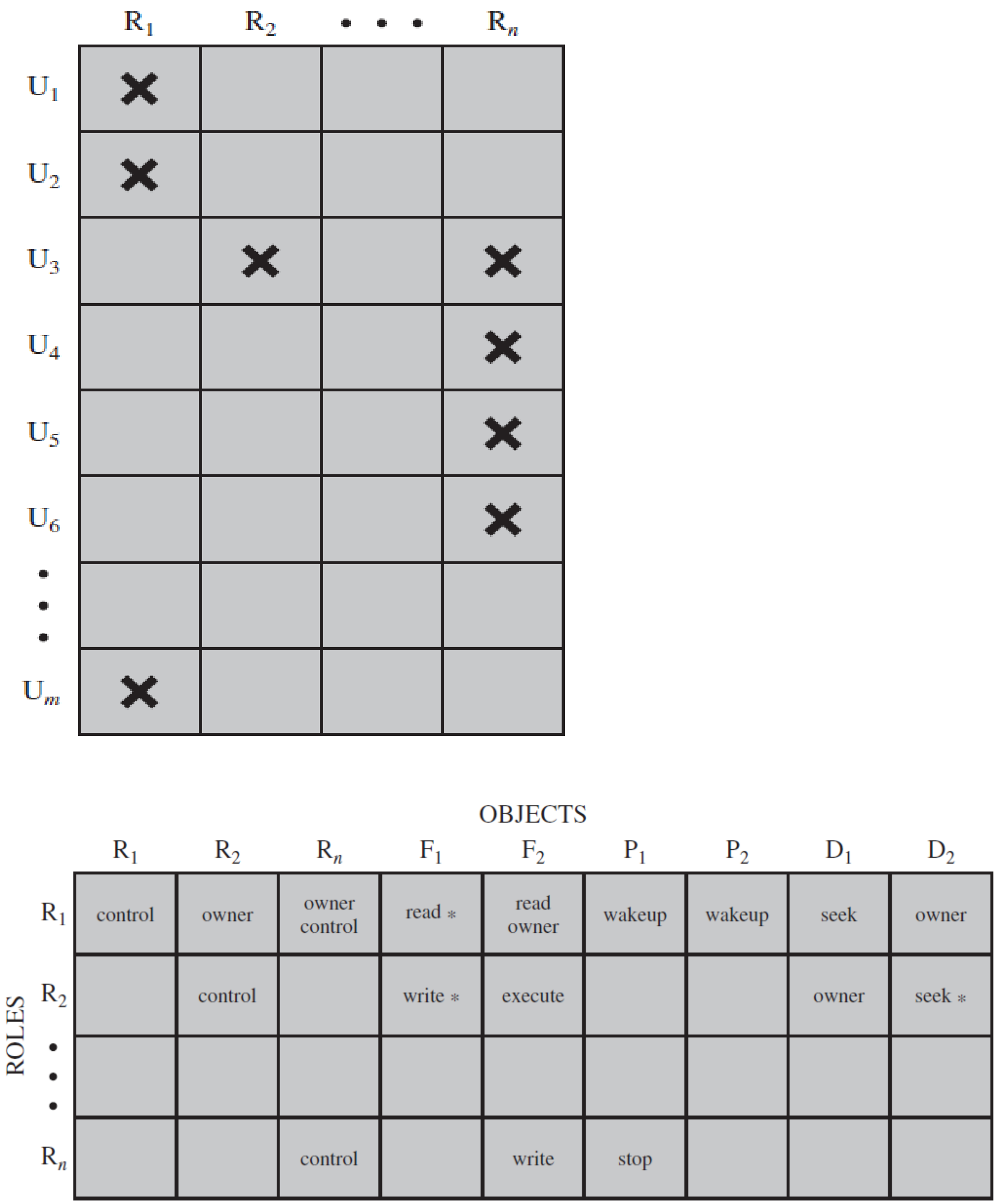
Role Based Access Control

- Traditional DAC system defines the access rights of individual users and groups. RBAC is based on
    - Roles that user assume in a system (instead of their identity)
    - Role is a job function within an organization. A role will have specific access rights to one or more resources
    - Assign Access rights to Roles (instead of individual users)

  - Users assigned to different roles according to their responsibilities
  - User-to-Roles are Many-to-Many

- The set of users change frequently, and the assignment of user to one or many roles is also dynamic
- The set of rules is relatively static, with only occasional addition or deletion.
- The set of Resources and the specific access right associated with a particular role are also likely to change infrequently which is relatively static.





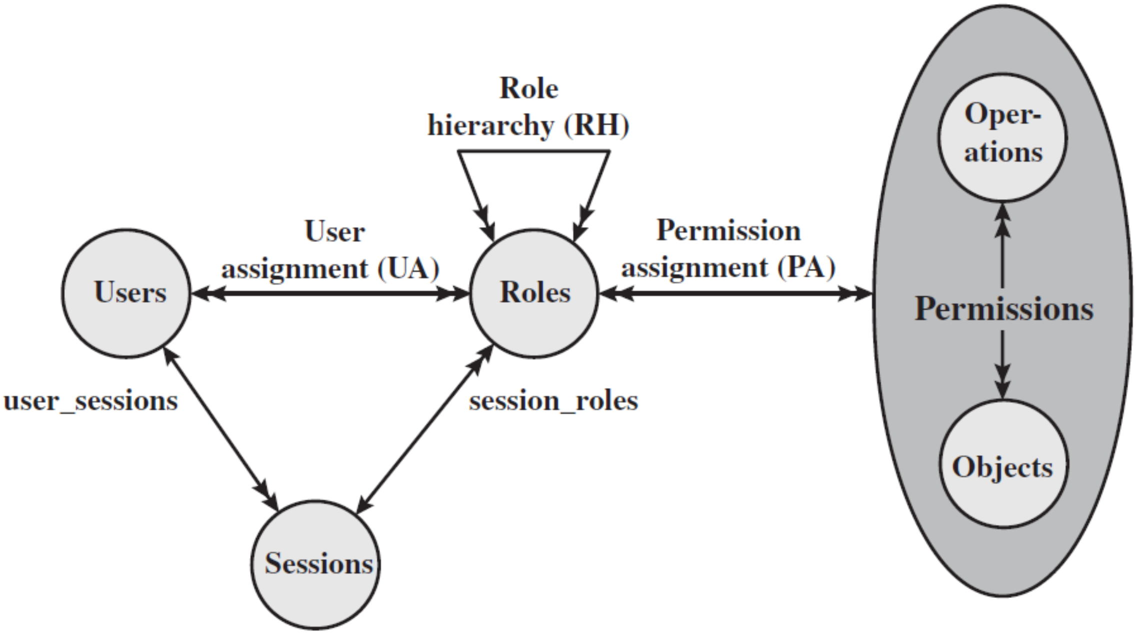
	R_1	R_2	• • •	R_n
U_1	×			
U_2	×			
U_3		×		×
U_4				×
U_5				×
U_6				×
• • •				
U_m	×			

OBJECTS

ROLES	R_1	R_2	R_n	F_1	F_2	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: Access Control Matrix Representation of RBAC

RBAC models:



The diagram represents a Role-Based Access Control (RBAC) model, which is used to manage access to resources within a system based on user roles. Here's a detailed explanation of each component and the relationships between them:

1. **Users**:
   - Represents the individuals who need access to the system.

2. **Roles**:
    - Represents a set of permissions or access rights within the system. Users are assigned to roles, and roles define what permissions the users have.
3. **Permissions**:
    - Represents the rights to perform certain operations on objects within the system. Permissions are granted to roles.
4. **Operations**:
    - Represents the actions that can be performed on objects (e.g., read, write, delete).
5. **Objects**:
    - Represents the resources or entities within the system on which operations are performed.
6. **Sessions**:
    - Represents the active instance of a user's interaction with the system. A session links a user to one or more roles.

Relationships:

1. **User Assignment (UA)**:
   - This arrow shows that users are assigned to roles. A user can be associated with one or multiple roles, and this assignment determines the user's access rights.
2. **Permission Assignment (PA)**:
   - This arrow shows that permissions are assigned to roles. A role can have one or more permissions, defining what actions users in that role can perform on objects.
3. **Role Hierarchy (RH)**:
   - This arrow indicates a hierarchy or inheritance relationship between roles. Higher-level roles inherit permissions from lower-level roles, allowing for organized and scalable permission management.
4. **User Sessions**:
   - This arrow indicates that users can have one or more active sessions.

  - Each session is associated with one or more roles that the user has been assigned.

5. **Session Roles**:
    - This arrow shows that sessions are linked to specific roles.
    - During a session, a user activates certain roles, which determines the permissions available during that session.

6. **Operations and Objects**:
    - The relationship between operations and objects defines what actions can be performed on which resources within the system.
    - This is encapsulated within the Permissions entity.

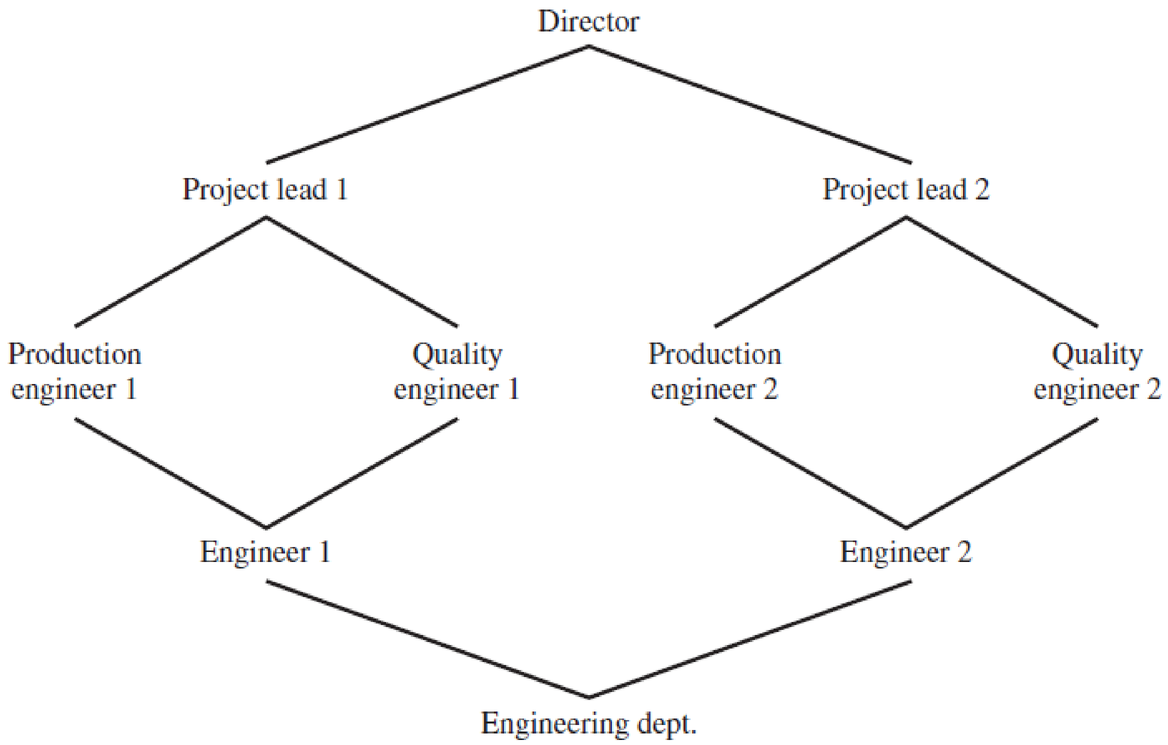


Figure: Example of Role Hierarchy

Attribute-Based Access Control

Attribute-Based Access Control (ABAC), also known as Policy-Based Access Control (PBAC) or Claim-Based Access Control (CBAC), is an advanced authorization methodology that sets and enforces policies based on various characteristics such as department, location, manager, and time of day.

Key Points of ABAC:

1. **Policy-Based and Claim-Based Control**:
   - ABAC policies are defined based on attributes or claims associated with users, resources, and environmental conditions.
   - Policies specify conditions under which access should be granted or denied.
2. **Use of Boolean Logic**:
   - ABAC policies use Boolean logic to define access control rules.
   - If-then statements specify access rights based on attribute values.
   - For example:

```
if (user.department == 'sales' && time_of_day == 'working_hours') {
    grant access to 'CRM';
}
```

3. **Example Scenario**:
   - Consider a CRM (Customer Relationship Management) system where access is controlled based on roles and other attributes:
     - **Salesperson**: If the requester is a salesperson, they are granted read-write access to the CRM solution.
     - **Administrator**: An administrator is only granted view privileges to create reports.
   - This is an example of a policy that differentiates access levels based on user roles and responsibilities.

Advantages of ABAC:

- **Fine-Grained Access Control**: ABAC allows for more detailed and specific access control compared to role-based models.
- **Dynamic Policies**: Policies can be dynamically evaluated based on real-time attributes and context, allowing for more responsive and adaptable access control.

- **Scalability**: ABAC is highly scalable as it does not rely on predefined roles and can accommodate a growing number of attributes and policies.
- **Flexibility**: It supports complex access control scenarios that involve multiple attributes and conditions.

Identity, Credentials, and Access management

Identity, Credentials, and Access management (ICAM) is a set of security tools, policies, and systems that helps organizations manage, monitor, and secure access to their organizations, IT infrastructure.

- ICAM represent the combination of digital identities, credentials, and access control into a single comprehensive approach.
- ICAM reduces the risk of cyber-attack to the organization by preventing unauthorized access to your network, system, and data

ICAM include:

Identity Management

Purpose: To identify a subject and establish their legitimacy for accessing a system, whether as a physical entity (e.g., a person) or a digital entity (e.g., an application or device).

Key Activities:

- **Identity Creation**: Establishing unique identities for subjects. This includes collecting and storing necessary attributes such as name, email, department, and role.
- **Identity Verification**: Ensuring that the subject is who they claim to be, typically during the initial registration or onboarding process. This might involve verifying documents or using biometric verification.
- **Identity Lifecycle Management**: Managing the lifecycle of identities, including onboarding, updating information as needed (e.g., changes in role or department), and offboarding when the identity is no longer needed.

Credential Management

Purpose: To bind the identity of the subject to an authenticator, allowing the system to identify the user through various methods of authentication.

Key Activities:

- **Credential Issuance**: Providing subjects with authenticators, which can include:
  - **User Authentication**: Methods like passwords, PINs, and multi-factor authentication.
  - **Identification Cards**: Physical or digital cards used for access.
  - **Biometric Data**: Fingerprints, facial recognition, or other biometric identifiers.
- **Credential Storage**: Securely storing credentials to prevent unauthorized access. This often involves encryption and secure storage solutions.
- **Credential Renewal and Revocation**: Managing the validity of credentials, including renewing them before they

expire and revoking them when they are no longer valid or have been compromised.

Access Management

Purpose: To grant permissions and manage what users can do and see within a system. Access management determines which roles or users can access different information and processes at specific times and levels of security.

Key Activities:

- **Role-Based Access Control (RBAC)**: Assigning access rights based on user roles within the organization. For example, a manager might have access to different resources compared to a regular employee.
- **Attribute-Based Access Control (ABAC)**: Using attributes (e.g., department, location, time of day) to determine access rights. Policies are defined using if-then statements to grant or deny access based on these attributes.

- **Permission Management**: Defining and managing permissions for resources. This includes creating groups and roles for separation of duties and ensuring users have the appropriate level of access.
- **Access Reviews and Audits**: Periodically reviewing access rights and conducting audits to ensure compliance with policies and detect any unauthorized access.

Trust Frameworks

Trust frameworks are structured sets of legal, business, technical, and operational requirements that define how entities within a network or ecosystem establish trust relationships and interact securely.

These frameworks provide a common foundation for trust, enabling different organizations or systems to interoperate smoothly and securely.

Trust frameworks are essential for creating consistent and reliable environments for digital transactions and interactions, especially in contexts like federated identity management, electronic commerce, and secure data sharing.

Components of Trust Frameworks

1. **Legal and Policy Requirements**:
    - **Agreements and Contracts**: Define the legal obligations and responsibilities of parties within the trust framework.
    - **Compliance and Regulation**: Ensure adherence to relevant laws, regulations, and industry standards (e.g., GDPR, HIPAA).
2. **Business Requirements**:
    - **Business Rules**: Outline the specific business practices and rules that participants must follow.
    - **Liability and Accountability**: Define liability in case of security breaches or non-compliance and establish accountability mechanisms.

3. **Technical Requirements**:
    - **Standards and Protocols**: Specify the technical standards and protocols that must be used for interoperability (e.g., SAML, OAuth, OpenID Connect).
    - **Security Measures**: Define security requirements such as encryption, authentication methods, and access control policies.
4. **Operational Requirements**:
    - **Operational Procedures**: Describe the procedures for managing identities, credentials, and access controls.
    - **Auditing and Monitoring**: Implement processes for continuous monitoring, auditing, and incident response to ensure compliance and security.