Access Control
Lesson Introduction

- Understand the **importance of access control**
- **Explore ways** in which access control can be implemented
- Understand **how access control** is implemented

Controlling Accesses to Resources

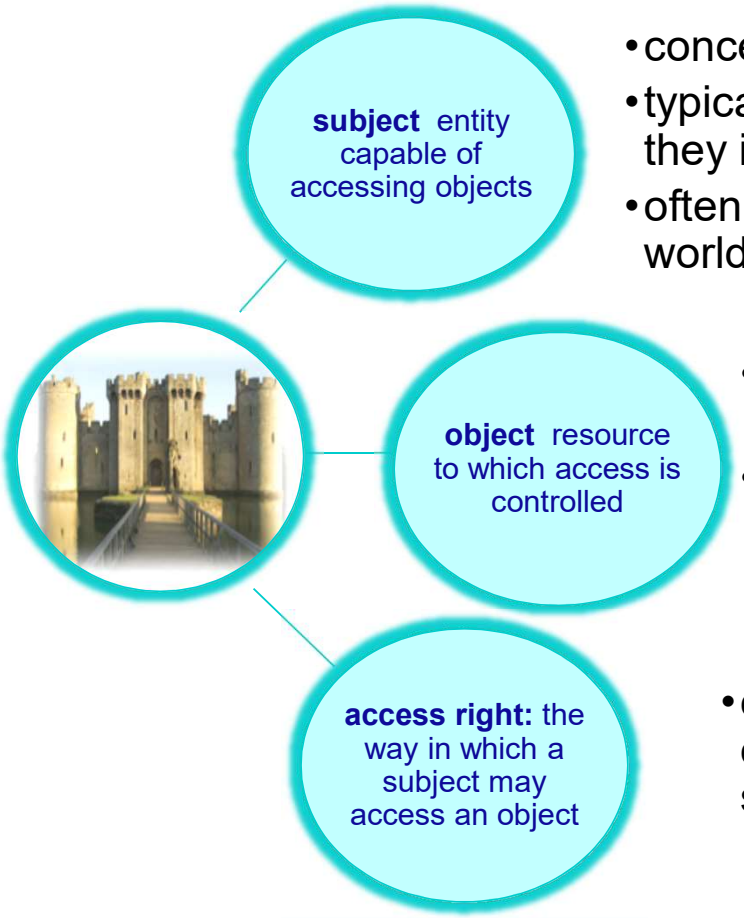
- **TCB** (reference monitor) sees a request for a resource, how does it decide whether it should be granted?

  - **Example:** Should John's process making a request to read a certain file be allowed to do so?

Controlling Accesses to Resources

- **Authentication** establishes the source of a request (e.g., John's UID)
- **Authorization** or access control answers the question if a certain source of a request (User ID) is allowed to read the file
- **Subject who owns a resource (creates it) should be able to control access to it (sometimes this is not true)**

Access Control Basic Elements



- concept equates with that of process
- typically held accountable for the actions they initiate
- often have three classes: owner, group, world

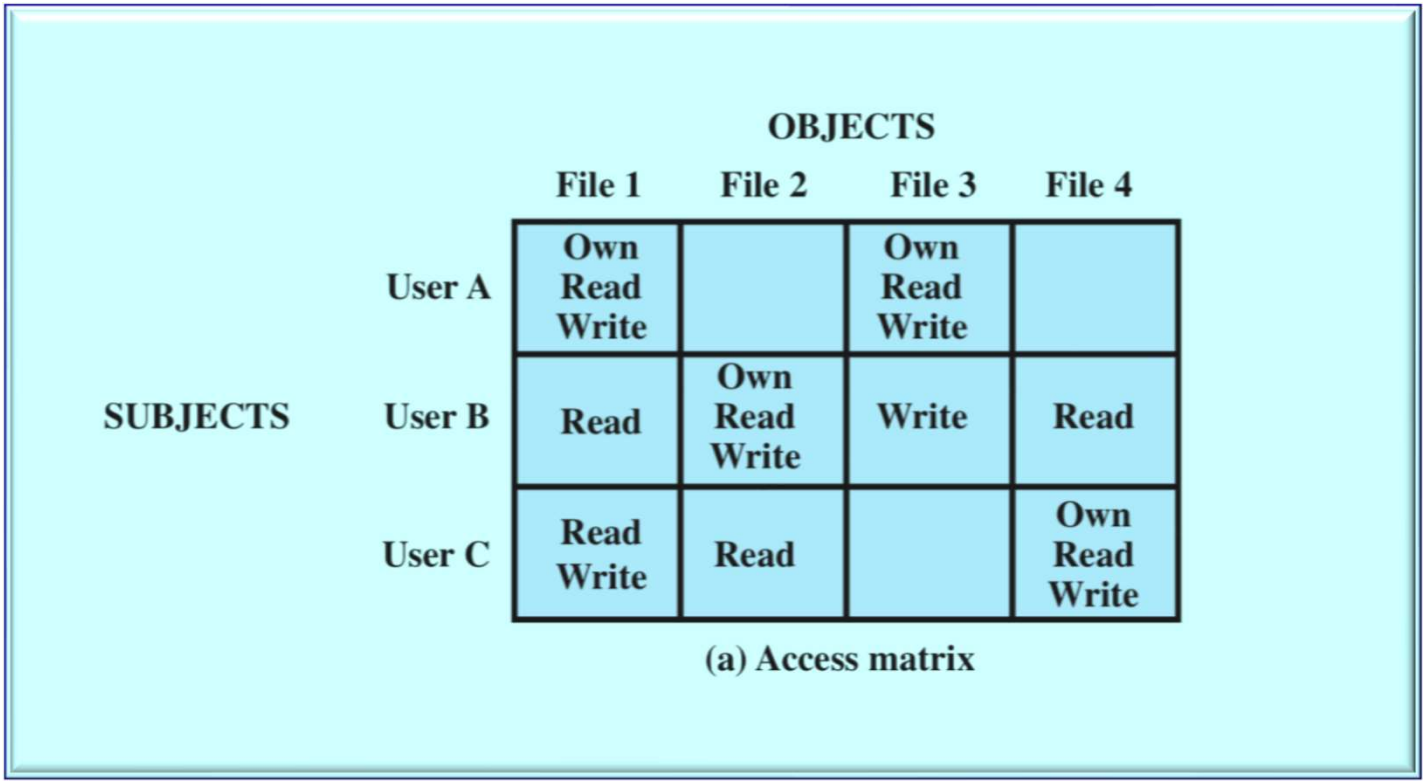
- entity used to contain and/or receive information
- protection depends on the environment in which access control operates

- e.g. read, write, execute, delete, create, search

Access Control Matrix (ACM)

- An access control matrix (ACM)
  abstracts the state relevant to access control.
- Rows of ACM correspond to users/subjects/groups
- Columns correspond to resources that need to be protected.
- **ACM defines who can access what**
  - ACM [U,O] define what access rights user U has for object O.

Access Matrix



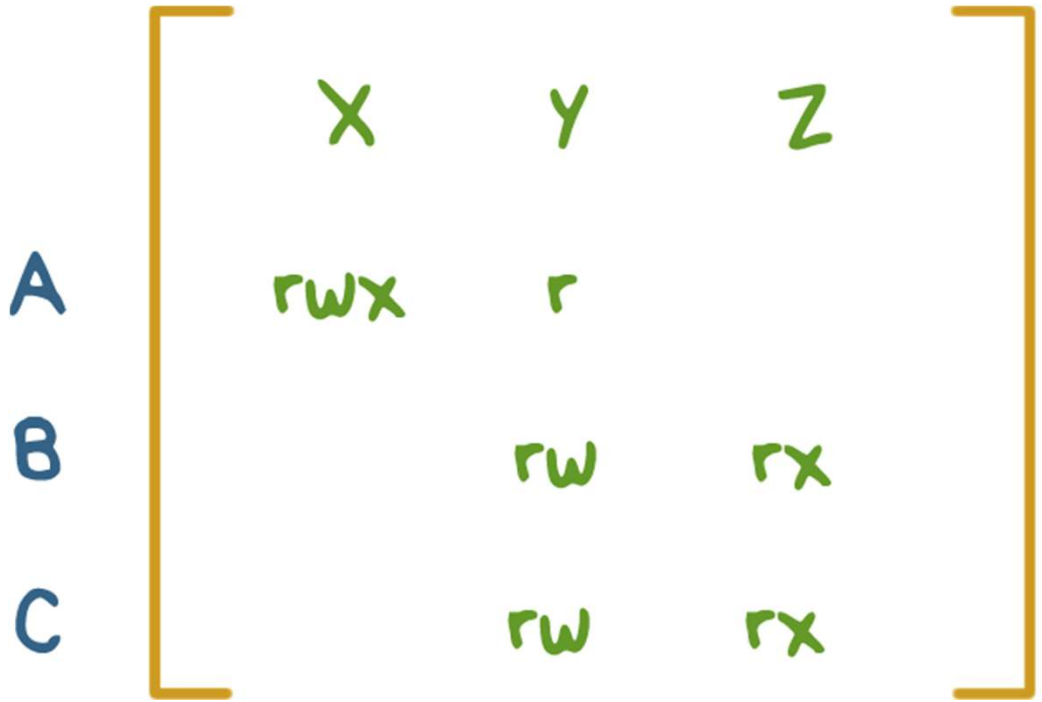
SUBJECTS	OBJECTS: File 1	File 2	File 3	File 4
User A	Own Read Write		Own Read Write	
User B	Read	Own Read Write	Write	Read
User C	Read Write	Read		Own Read Write

(a) Access matrix

Implementing Access Control

- **Access control matrix is large**
- How do we represent it in the system?
  - Column for object Oi is [(ui1, rights1), (ui2, rights2),…]
    - Called **access control list or ACL**
    - Associated with each resource
  - For user ui, a row in the matrix is [(oi1, rights1), (oi2,rights2,….].
    - Called a capability-list or C-list.
    - Such a C-list stored for each user

Implementing Access Control



ACLs

X → [(A,rwx)]
Y → [(A,r)(B,rw)(C,rw)]
Z → [(B,rx)(C,rx)]

C-lists

A → [(X,rwx)(Y,r)]
B → [(Y,rw)(Z,rx)]
C → [(Y,rw)(Z,rx)]

ACL and C-Lists Implementation:

- **Where should an ACL be stored?**
  - In trusted part of the system
  - Consists of access control entries, or, ACEs
  - Along with other object meta-data
  - For example, file meta-data has a bunch of information where this can go as well
  - Checking requires traversal of the ACL

ACL and C-Lists Implementation:

C-list

- **Where do C-lists go?**
  - A capability is an unforgeable reference/handle for a resource
  - User catalogue of capabilities defines what a certain user can access
  - Can be stored in objects/resources themselves (Hydra)
  - Sharing requires propagation of capabilities

ACL and C Lists Implementation:

ACL vs. **C-list**

Efficiency

C-list

Accountability

ACL

Revocation

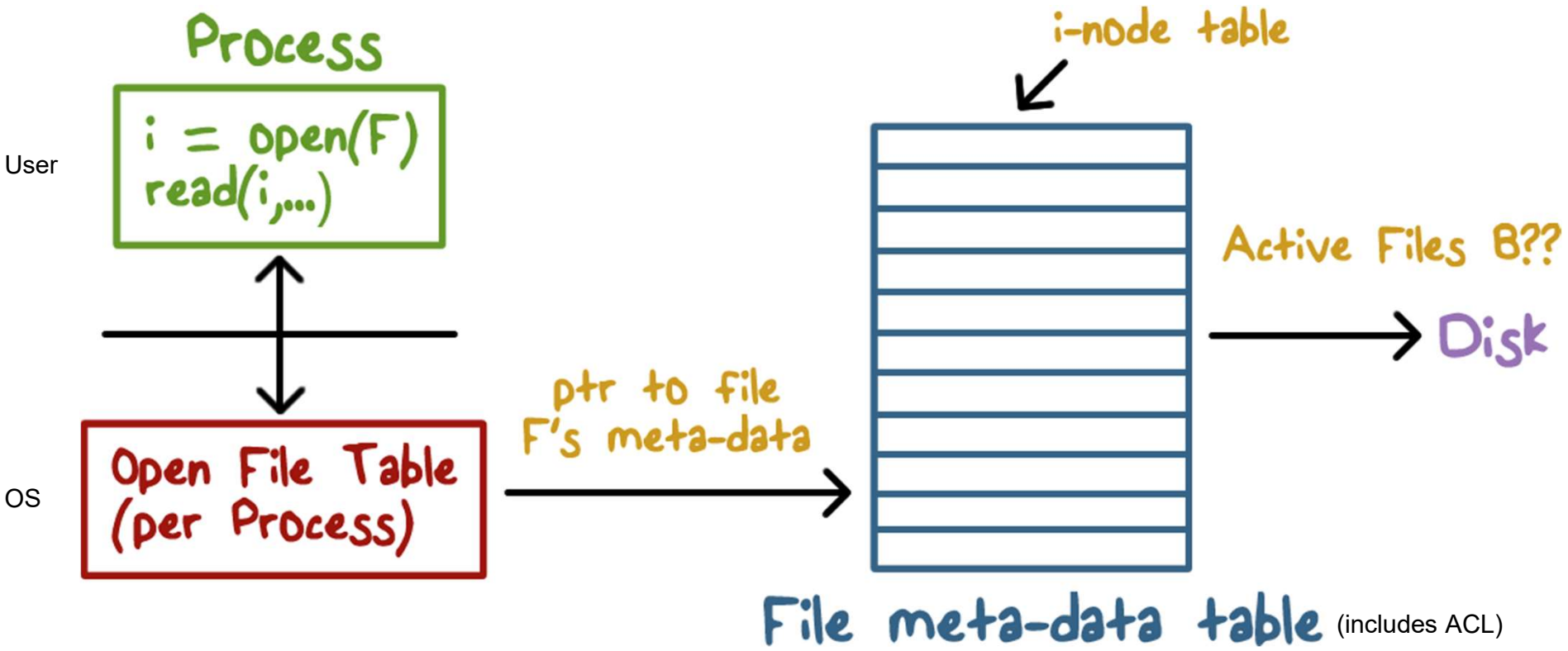
ACL

Access Control Implementation

How is Access Control Implemented in Unix-like Systems?

- In Unix, **each resource looks like a file**.
- Each file has an owner (UID) and access is possible for owner, group and everyone (world).
- **Permissions are read, write and execute.**
- Original ACL implementation had a compact fixed size representation (9 bits)
- **Now full ACL support is available in many variants** (Linux, BSD, MacOS,..)
- Few other things (sticky bit, setuid,…)

How does the OS Implement ACL?



Time to Check vs. Time to Use (TOCTOU)

A time-to-check-time-to-use vulnerability arises when access check is performed separately from when a file is read or written. TOCTOU vulnerability arises when file permissions change after an open() call completes for the file and before it is closed.

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
- Mandatory access control (MAC)
  - Controls access based on comparing security labels with security clearances
- 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

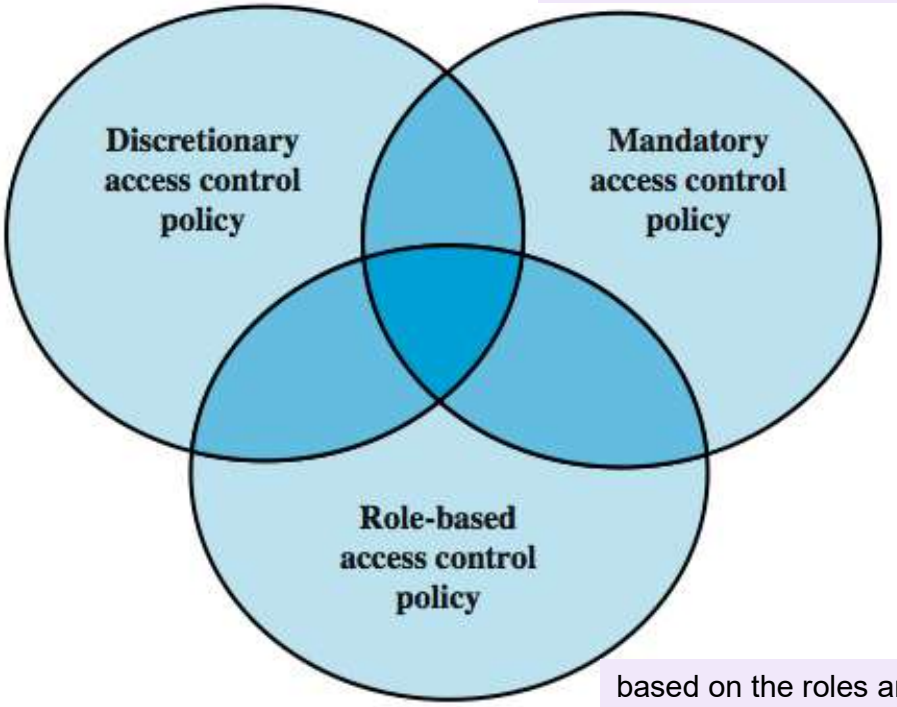
Access Control Policies

Dictates

- what types of access are permitted,
- under what circumstances,
- by whom.

based on comparing security labels with clearances

based on the identity of the requestor and on access rules



based on the roles and their accesses

Discretionary Access Control

- 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

Role-Based Access Control (RBAC)

- In enterprise setting, **access may be based on job function or role of a user**
  - Payroll manager, project member etc.
  - Access rights are associated with roles
- **Users authenticate themselves** to the system
- **Users then can activate one or more roles** for themselves

RBAC Benefits

- Policy **need not be updated when a certain person with a role leaves** the organization
- New employee **should be able to activate the desired role**
- **Revisiting least privilege**
  - User in one role has access to a subset of the files
  - Switch roles to gain access to other resources

Access Control
Lesson Summary

- Fundamental requirement when **resources need to be protected**
- An **access control matrix** captures who can access what and the manner in which it can be done
- **ACLs and C-lists** are ways for implementing access control
- Getting **access control policy right is challenging**