Access Control

Elisa Bertino
CERIAS and CS &ECE Departments
Purdue University

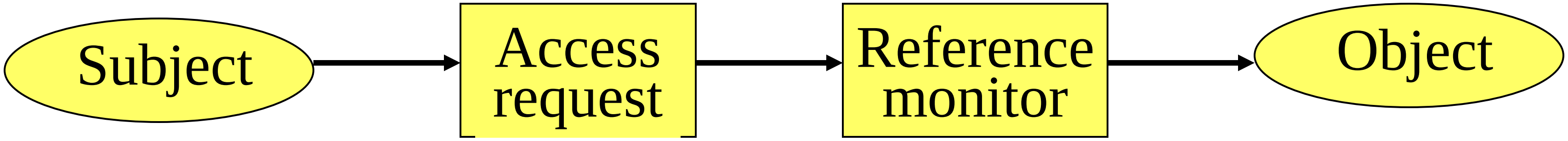
Access Control - basic concepts

- An access control system regulates the operations that can be executed on data and resources to be protected

- Its goal is to control operations executed by subjects in order to prevent actions that could damage data and resources

- Access control is typically provided as part of the operating system and of the database management system (DBMS)

Access Control - basic concepts



- The very nature of access control suggests that there is an *active* subject *requiring access* to a passive *object* to perform some specific *access operation.*
- A *reference monitor* grants or denies access
- This fundamental and simple notion of access control is due to Lampson

----

B. Lampson. Protection. *ACM Operating System Reviews*, **8**, 1974.

Access Control Mechanism

- It is typically a software system implementing the access control function
- It is usually part of other systems
- The access control mechanism uses some access control policies to decide whether to grant or deny a subject access to a requested resource
- We will refer to an *access control system* as system comprising an access control mechanism and all information required to take access control decisions (for example, access permissions)

Object

- Anything that holds data, such as relations, directories, interprocess messages, network packets, I/O devices, or physical media
- We often refer to objects, controlled by the access control system, as *protection objects*
- Note that not all resources managed by a system need to be protected

Subject

- An abstraction of any active entity that performs computation in the system
- Subjects can be classified into:
  - *users* -- single individuals connecting to the system
  - *groups* -- sets of users
  - *roles* -- named collections of privileges / functional entities within the organization
  - *processes* -- executing programs on behalf of users
- Relations may exist among the various types of subject

Access Operations - Access Modes

- Operations that a subject can exercise on the protected objects in the system
- Each type of operation corresponds to an ***access mode***
- The basic idea is that several different types of operation may be executed on a given type of object; the access control system must be able to control the specific type of operation
- The most simple example of access modes is:

– read	look at the contents of an object
– write	change the contents of an object

- In reality, there is a large variety of access modes: the access modes supported by an access control mechanism depend on the resources to be protected (read, write, execute, select, insert, update, delete, …)
- Often an access control system uses modes with the same name for different types of object; the same mode can correspond to different operations when applied to different objects

Access Operations - Access Modes An example

- Unix operating system
  - Access modes defined for files
    - read: reading from a file
    - write: writing to a file
    - execute: executing a (program) file
  - Access models defined for directories
    - read: list a directory contents
    - write: create or rename a file in a directory
    - execute: search a directory

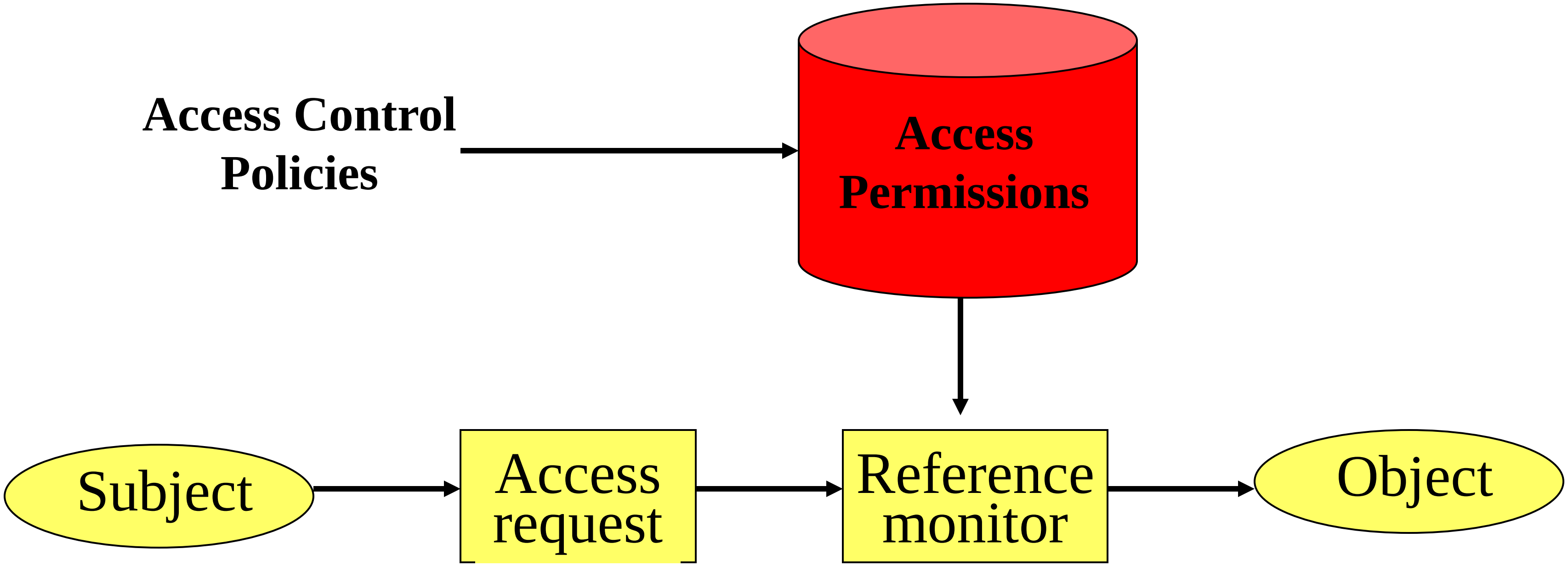
Access Operations

Access Permissions and Attributes

- How does the reference monitor decides whether to give access or not?
- Main approaches:
  - It uses *access permissions*
    - Typical of discretionary access control (DAC) models
  - It uses information (often referred to as *attributes*) concerning subjects and objects
    - Typical of multilevel access control (MAC) models
- More innovative approaches have been developed where access permissions can be also expressed in terms of object and subject attributes and even context parameters

Access Operations

Access Permissions



Access Permissions

- Access permissions, also called *authorizations*, are expressed in terms of subjects, objects, and access modes
- From a conceptual point of view an access permission is a tuple <*s*, *o*, *a*> where
  - *s* is a subject
  - *o* is an object
  - *a* is an access mode

  It states that subject *s has the permission* to execute operation *a* on object *o*

  We also say that *s has access right a* on object *o*
- Example: the access permission <Bob, F1, Read> states that Bob has the permission to read file F1

Access Permissions

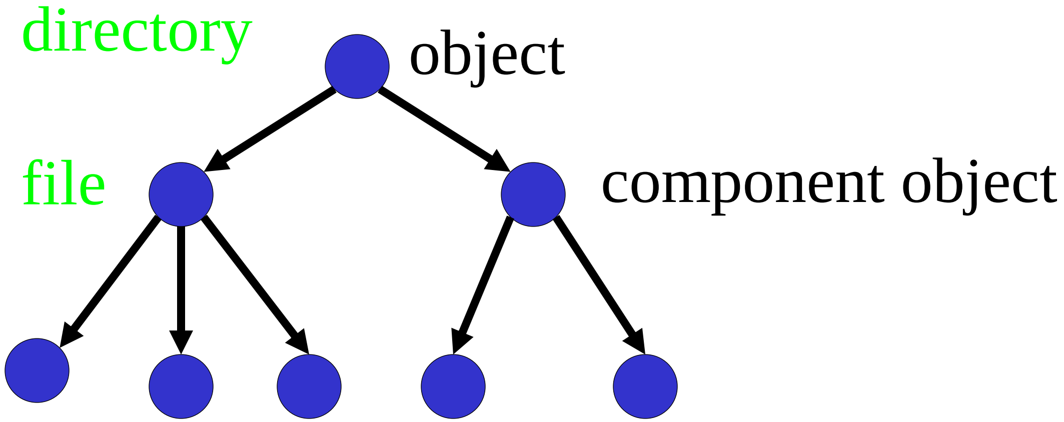
- Subjects, objects, and access modes can be organized into hierarchies

- The semantics of the hierarchy depends on the domain

- The use of hierarchies has two important advantages:
  - It reduces the number of permissions that need to be entered into the access control system, thus reducing administration costs and errors
  - Combined with negative authorizations (to be discussed later on), it supports the specification of exceptions

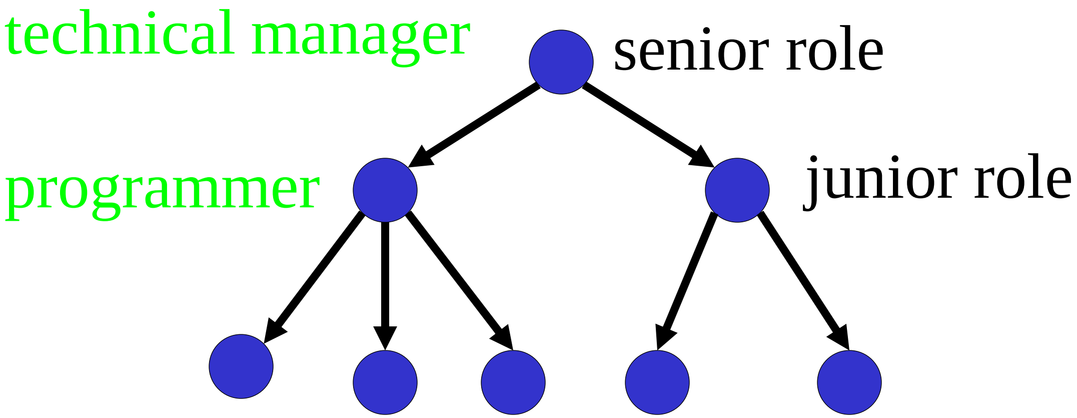
Object Hierarchy

PART-OF



In most cases all rights of a directory are also rights of the file in the directory (but not vice versa)

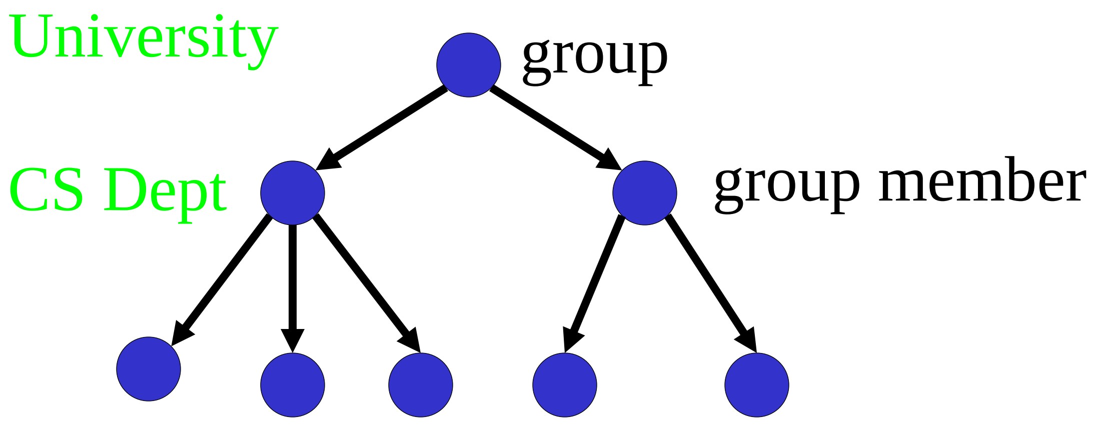
Role Hierarchy



In most cases all rights of a junior role are also rights of the corresponding senior role (but not vice versa)

Group Hierarchy

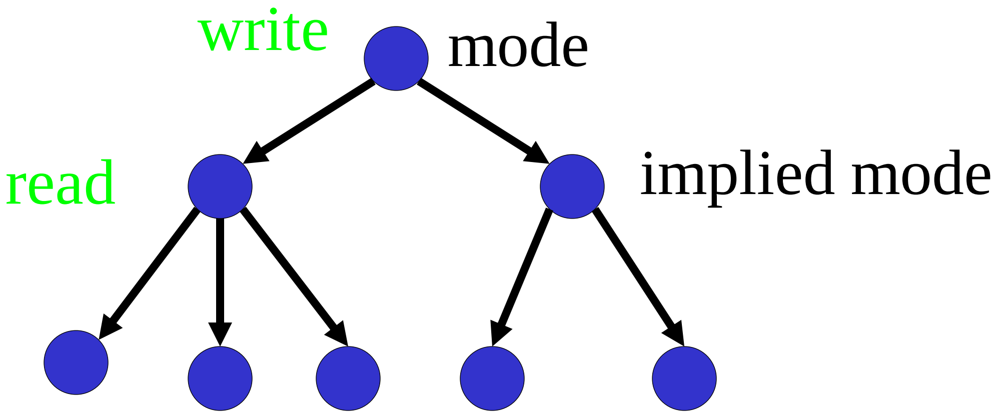
GROUP MEMBERSHIP



Suppose that the group CS department has 200 members and the University group 5000 members; suppose we have the policy that the department calendar can be read to all members of the University and written only by the members of CS; these policies can be encoded into two access permissions of the form:
<University, calendar, Read>   <CS Dept, calendar, Write>

Access Mode Hierarchy

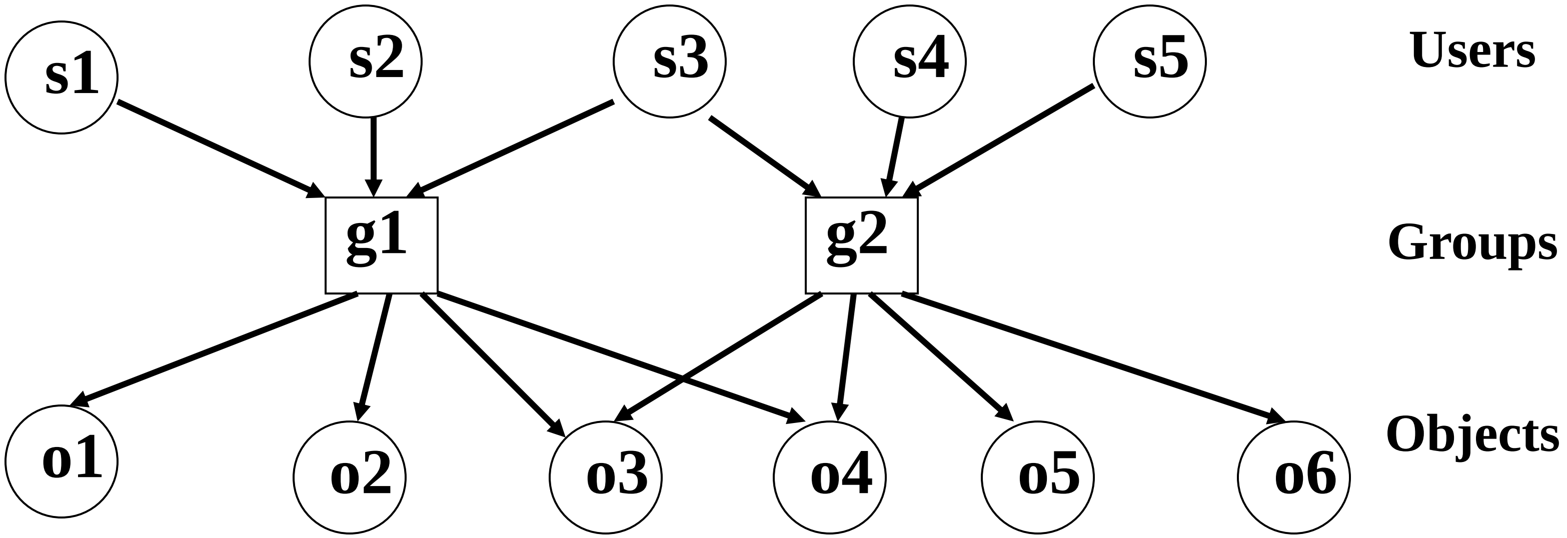
SUBSUMPTION



If a subject is allowed to write an object that he should be able to read it

Groups and Negative Permissions

- Groups can be seen as an intermediate level between users and objects
- An example of an ideal world where all access permissions are mediated by groups



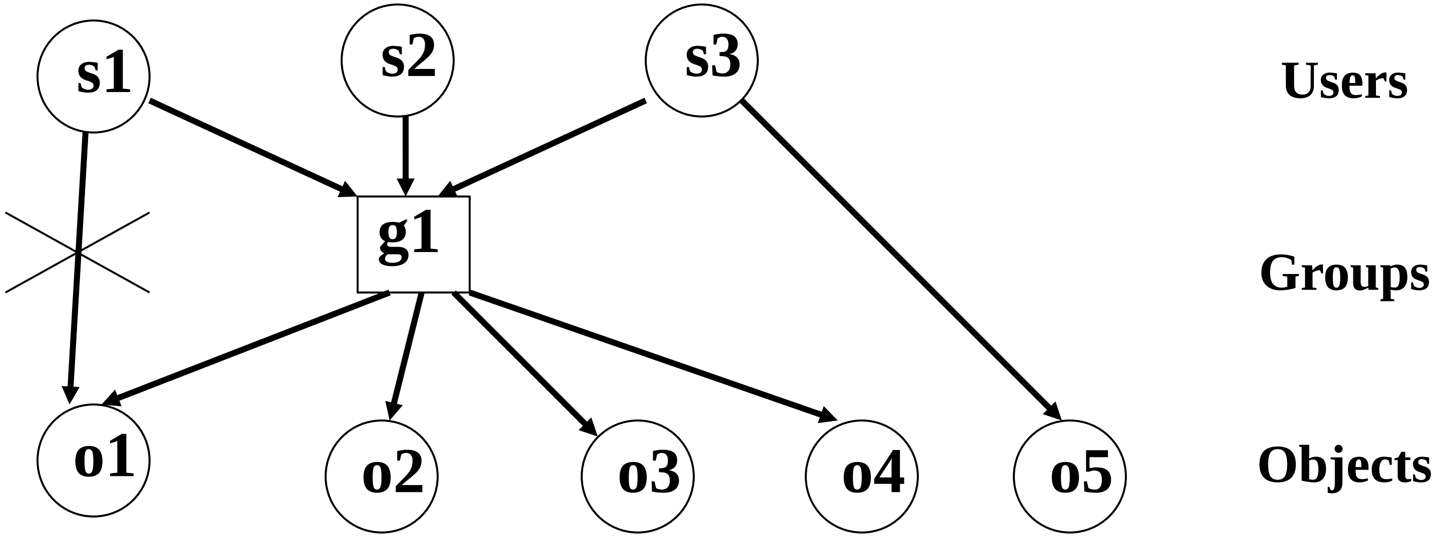
Groups and Negative Permissions

- Often access control policies have special cases where it is proved convenient to give some user a permission for an object directly or *deny* a user a permission that it would normally derive from its membership in some group
- A *negative* permission specifies an operation that a subject is not allowed to perform
- Representing negative permissions requires extending our simple tuple model with an additional component:

  $\langle s, o, a, sign \rangle$ where $sign \in \{+, -\}$

Groups and Negative Permissions

An example in which not all access permissions are mediated through groups



Ownership and Administration

- A key question when dealing with access control is who specifies which subjects can access which objects for which operations
- In the case of permissions, this means specifying which are the subjects that can enter permissions

Ownership and Administration
Two basic options

- *Discretionary* approach
  - the owner of a resource decrees who is allowed to have access
  - But then: who is the owner of a resource?
- *Mandatory* approach
  - a system-wide policy decrees who is allowed to have access
- These approaches are the conventional ones
  - today we need more sophisticated approaches
  - (cf. the column M. Donner “Whose Data are These, Anyway”, *Security&Privacy*, May-June 2004)

Access Control Structures

The best known access control structures for DAC models are based on the notion of *Access Control Matrix*. Let:

- *S* be a set of subjects
- *O* be a set of objects
- *A* be a set of access modes

An access control matrix *M* on *S*, *O*, and *A* is defined as

$$M = (M_{so})_{s \in S,\ o \in O} \text{ with } M_{so} \in A$$

The entry M_{so} specifies the set of access operations subject *s* can perform on object *o*.

Access Control Structures Example

	bill.doc	**edit.exe**	**fun.dir**
Alice	**-**	{execute}	{execute, read}
Bill	{read, write}	**-**	{execute, read, write}

Access Control Structures

Access Control Lists and Capabilities

- Directly implementing access control matrices is quite inefficient, because in most cases these matrices are sparse
- Therefore two main implementations have been developed
  - Access control lists
    - Used in DBMS and Operating Systems
  - Capabilities

Basic Operations in Access Control

- *Grant* permissions
  - – Inserting values in the matrix's entries
- *Revoke* permissions
  - – Remove values from the matrix's entries
- *Check* permissions
  - – Verifying whether the entry related to a subject *s* and an object *o* contains a given access mode