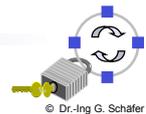


Network Security

Chapter 9 Access Control

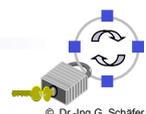
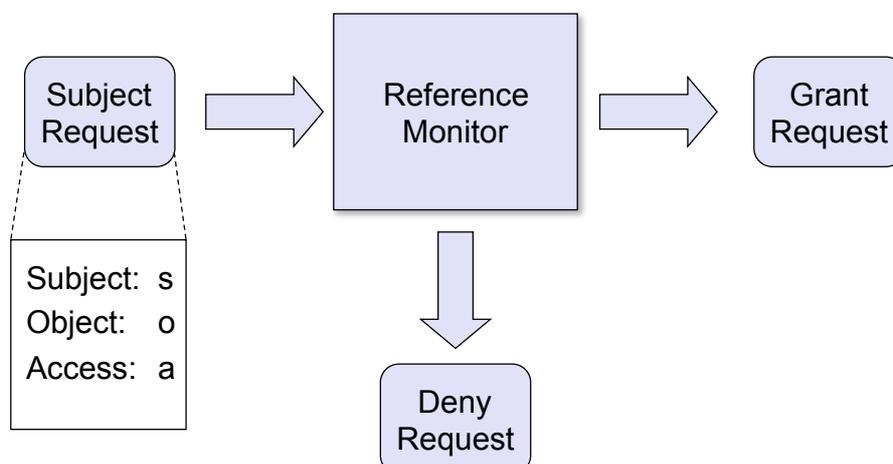


What is Access Control?

□ Definition:

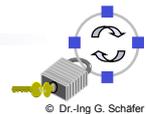
Access control comprises those mechanisms that enforce mediation on subject requests for access to objects as defined in some specified security policy.

□ An important conceptual model in this context is the *reference monitor*.



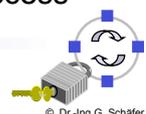
Security Policy (with respect to access control)

- ❑ In order to make access control decisions, the reference monitor needs to know the *security policy* of the system
- ❑ Definition:
The *security policy* of a system defines the conditions under which subject accesses to objects are mediated by the system reference monitor functionality
- ❑ Remarks:
 - ❑ The above definition is usually given in the context of computer and operating systems security
 - ❑ The reference monitor is just a conceptual entity, it does not necessarily need to have a physical or logical counterpart in a given system
 - ❑ The term *security policy* is often also used in a wider sense to describe a specification of all security aspects of a system including threats, risks, security objectives, countermeasures, etc.



Classical Computer Subjects, Objects & Types of Access

- ❑ Definition:
A *subject* is an active entity that can initiate a request for resources and utilize these resources to complete some task
- ❑ Definition:
An *object* is a passive repository that is used to store information
- ❑ The above two definitions come from classical computer science:
 - ❑ Subjects are processes, and files, directories, etc. are objects
- ❑ However, it is not always obvious to identify subjects and objects in the context of communications:
 - ❑ Imagine an entity sending a message to another entity: is the receiving entity to be viewed as an object?
- ❑ Furthermore, we need to have some understanding of what is an *access* and what types of access do exist:
 - ❑ Classical computer science examples for access types: read, write, execute
 - ❑ Object oriented view: any method of an object defines one type of access



Security Labels (1)

□ Definition:

A *security level* is defined as a hierarchical attribute with entities of a system in order to denote their degree of sensitivity

□ Examples:

- Military: unclassified < confidential < secret < top secret
- Commercial: public < sensitive < proprietary < restricted

□ Definition:

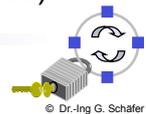
A *security category* is defined as a nonhierarchical grouping of entities to help denote their degree of sensitivity

- Example (commercial): department A, department B, administration, etc.

□ Definition:

A *security label* is defined as an attribute that is associated with system entities to denote their hierarchical sensitivity level and security categories

- In terms of mathematical sets: Labels = Levels \times Powerset(Categories)



Security Labels (2)

□ Security labels that denote the security sensitivity of:

- Subjects are called *clearances*
- Objects are called *classifications*

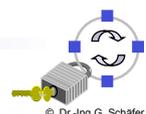
□ An important concept to the specification of security policies are *binary relations* on the set of labels:

- A binary relation on a set S is a subset of the cross-product $S \times S$
- Example:

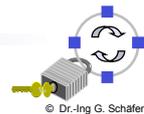
- *Dominates: Labels \times Labels*

$$\text{Dominates} = \{(b1, b2) \mid b1, b2 \in \text{Labels} \wedge \\ \text{level}(b1) \geq \text{level}(b2) \wedge \\ \text{categories}(b2) \subseteq \text{categories}(b1)\}$$

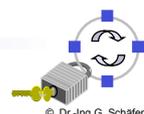
- If $(b1, b2) \in \text{Dominates}$, we also write *b1 dominates b2*



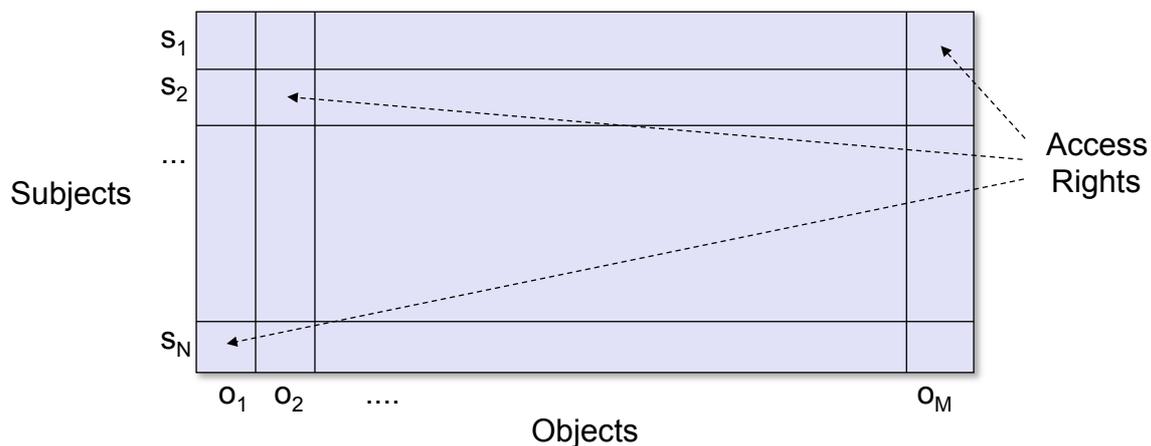
- Formal expressions for security policy rules:
 - Consider the following mappings:
 - allow: Subjects \times Accesses \times Objects \rightarrow boolean
 - own: Subjects \times Objects \rightarrow boolean
 - admin: Subjects \rightarrow boolean
 - dominates: Labels \times Labels \rightarrow boolean
 - The above mappings can be used to specify well-known security policies:
 - ownership: $\forall s \in \text{Subjects}, o \in \text{Objects}, a \in \text{Accesses}$:
allow(s, o, a) \Leftrightarrow own(s, o)
 - own_admin: $\forall s \in \text{Subjects}, o \in \text{Objects}, a \in \text{Accesses}$:
allow(s, o, a) \Leftrightarrow own(s, o) \vee admin(s)
 - dom: $\forall s \in \text{Subjects}, o \in \text{Objects}, a \in \text{Accesses}$:
allow(s, o, a) \Leftrightarrow dominates(label(s), label(o))
- The dom-policy requires a system to store and process security labels for each entity, but allows for more complex access control schemes than the ownership and own_admin policies



- An *access control mechanism* is an actual realization of the reference monitor concept
- There are two main types of access control mechanisms:
 - *Discretionary access control* comprises those procedures and mechanisms that enforce the specified mediation at the discretion of individual users
 - Example: the Unix operating system allows users to give or withdraw the read/write/execute access rights for files they own
 - *Mandatory access control* comprises those procedures and mechanisms that enforce the specified mediation at the discretion of a centralized system administration facility
- Both types may be combined, with the mandatory access control decisions most of the times overriding discretionary ones
 - Example:
 - Use of discretionary access control on personal computers combined with mandatory access control for communications (\rightarrow firewalls)



- A useful concept in the description of access control mechanisms is the *access matrix*:
 - In an access matrix for two sets of subjects and objects every row corresponds to one subject and every column to one object
 - Each cell of the matrix defines the access rights of the corresponding subject to the corresponding object



- **Access Control Lists (ACL):**
 - ACLs are the basis for an access control scheme, where for each object a list of valid subjects is stored which might have access to this object (possibly together with the type of access that is allowed)
 - ACLs are usually used with discretionary access control, as there are too many ACLs for being maintained by a central administration facility
- **Capabilities:**
 - Capabilities are somehow the opposite concept to ACLs as with capabilities each subject owns a list of access rights to objects
 - The advantage (and danger) of capabilities is, that a subject can give some of it's capabilities to other subjects
- **Label-based access control:**
 - If security labels are stored and processed with the entities of a system, they can be used to perform label-based access control
 - This scheme is usually used as a mandatory access control mechanism

→ Data integrity of access control data structures is critical!