INF3510 Information Security
University of Oslo
Spring 2010

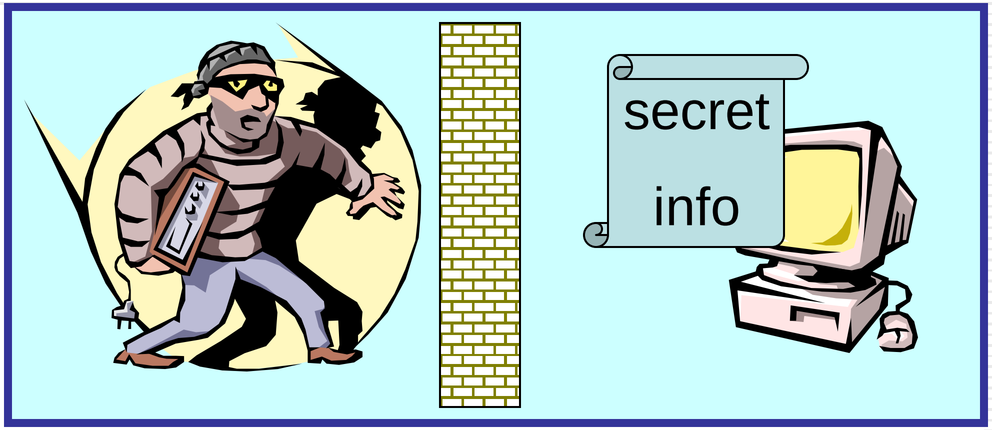
Lecture 5

Access Control and Security Models

Audun Jøsang

Introduction to access control

Physical Access Control:



Logical Access Control:

(theme for this lecture)

Introduction to access control

- Access Control
  - controls how users and systems access other systems and resources
  - prevents unauthorized users access to resources
  - prevents authorized users from misusing resources
- Some information assets may be accessible to all, but access to some information assets should be restricted.
- Unauthorized access could compromise
  - Confidentiality
  - Integrity
  - Availability

  of information assets

Basic concepts

- Access control philosophies:
  - ***Who should have access to resources?***
- Access requests can be:
  - generally permitted unless expressly forbidden (example: blacklist)
    - If your name is on the list, you will be denied access
  - generally forbidden unless expressly permitted (example: least privilege, need to know)
    - user access restricted to resources they need to perform their day-to-day business function, and nothing more
    - This is generally more secure

Basic concepts

- Access control philosophies continues:
- Separation of privileges:
  - A subject should not be able to execute a highly critical task alone
    - More than one person is required to complete the task
    - E.g. Financial transactions may require authorisation of two users
  - Conflict of interest should be avoided
    - E.g. A lawyer should not handle two sides of the same case, or handle the cases of competitors

Basic concepts

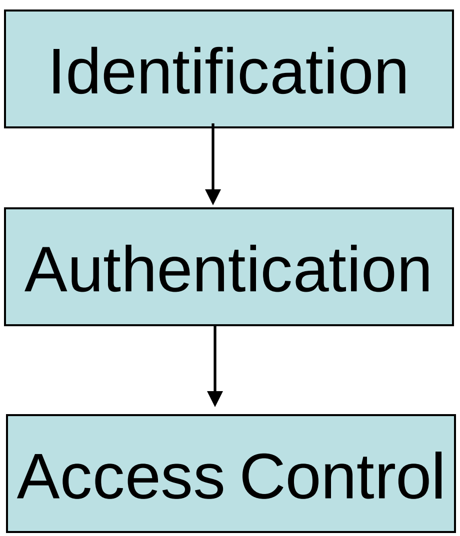
- Access control terminology:
  - Subjects
    - Entities requesting access to a resource
    - Examples: Person (User), Process, Device
    - Active
    - Initiate the request and is the user of information
  - Objects
    - Resources or entities which contains information
    - Examples: Disks, files, records, directories
    - Passive
    - Repository for information, the resources that a subject tries to access

Basic concepts

- Modes of access:
  - ***What access permissions (authorizations) should subjects have?***
- If you are authorized to access a resource, what are you allowed to do to the resource?
  - Example: possible access permissions include
    - read - observe
    - write – observe and alter
    - execute – neither observe nor alter
    - append - alter

Basic Concepts

- Sequence of Identification, authentication and access control



Who are you?

Is it really you?

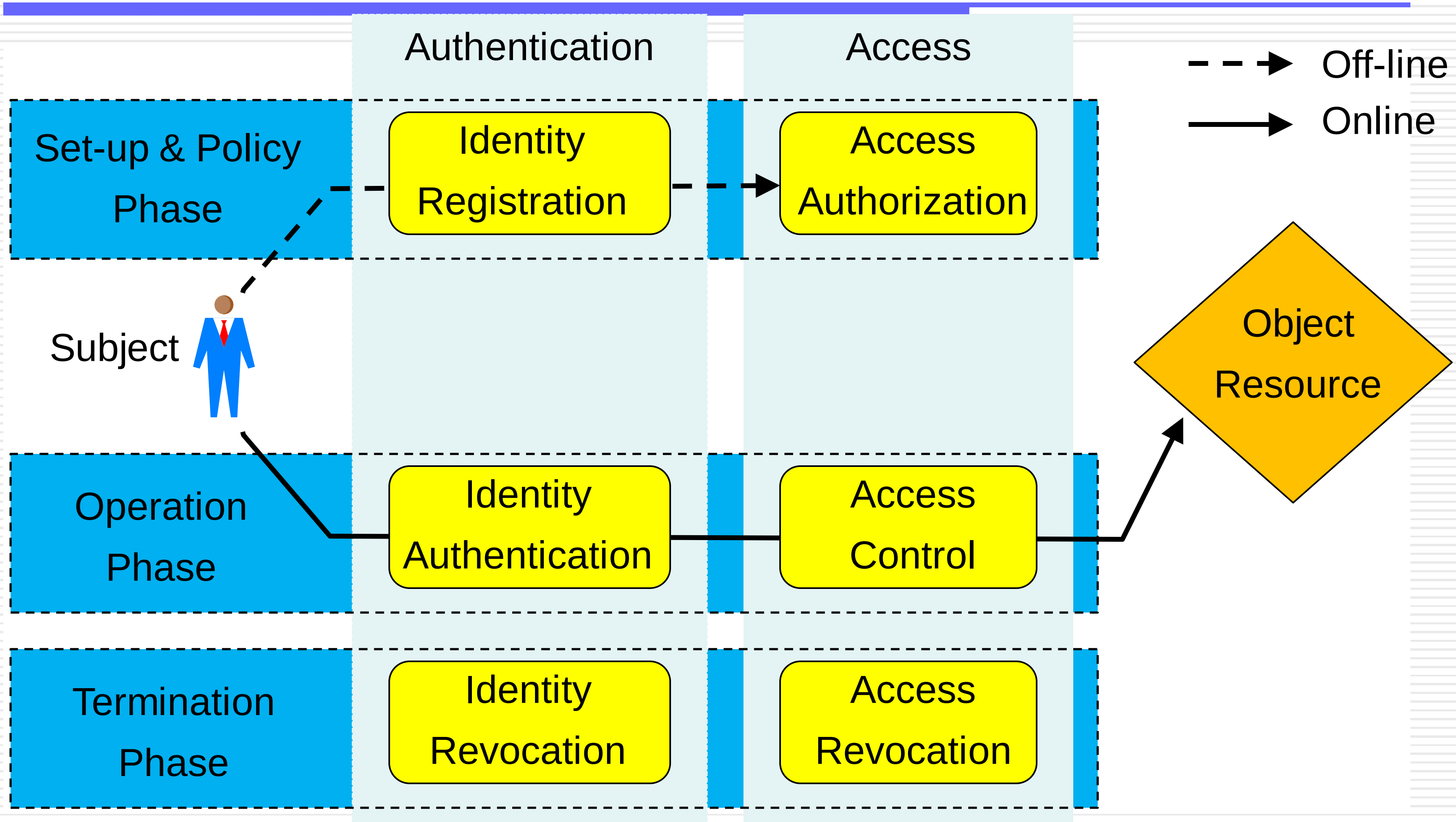
Are you authorized to access this resource?

Basic concepts

- Three phases of access control
  1. Policy definition (authorization) phase
     a. Authorise subject by defining the AC policy
     b. Distribute access credentials/token to subject
     c. Change authorization whenever necessary
  2. Policy enforcement (grant access) phase*
     a. Authenticate subject
     b. Grant access as authorised by policy
     c. Monitor access
  3. Termination phase

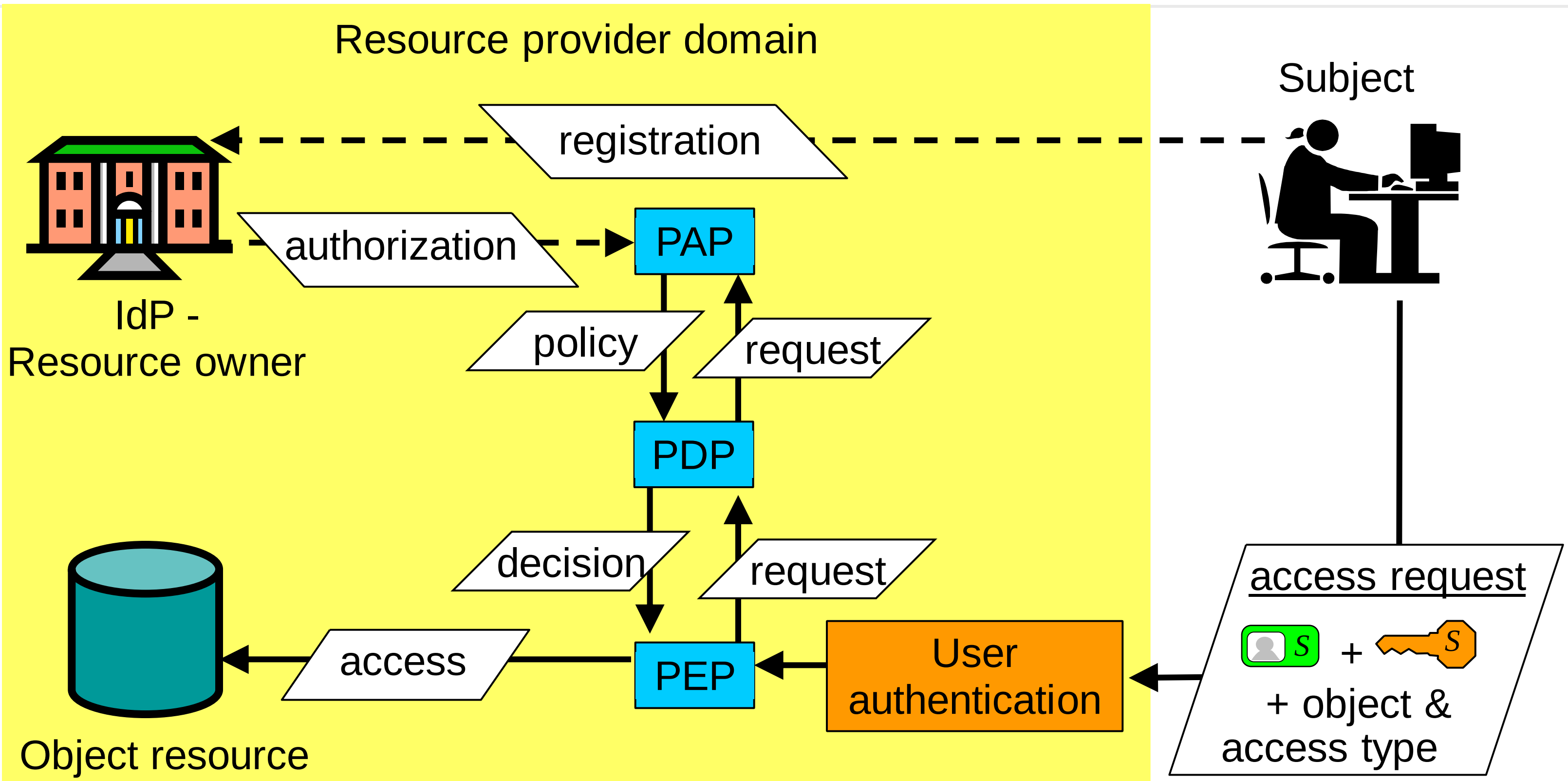
     De-register identity / Revoke authorization

Authentication and Access phases



Access control conceptual diagram

WS-Security terminology and architecture (http://www.oasis-open.org/specs/index.php)



PAP: Policy Administration Point

PDP: Policy Decision Point

PEP: Policy Enforcement Point

IdP: Identity Provider

Off-line

Online

Basic concepts

- Access control approaches:
  - *How do you define which subjects can access which objects?*
- Three main approaches
  - Discretionary access control (DAC)
  - Mandatory access control (MAC)
  - Role-based access control (RBAC)

Basic concepts: DAC

- Discretionary access control
  - Access rights to an object or resource are granted at the discretion of the owner
    - e.g. security administrator, the owner of the resource, or the person who created the asset
  - DAC is discretionary in the sense that a subject with a certain access authorization is capable of passing that authorization (directly or indirectly) to any other subject.
  - Usually implemented with ACL (Access Control Lists)
  - Popular operating systems use DAC.

Basic concepts: ACL

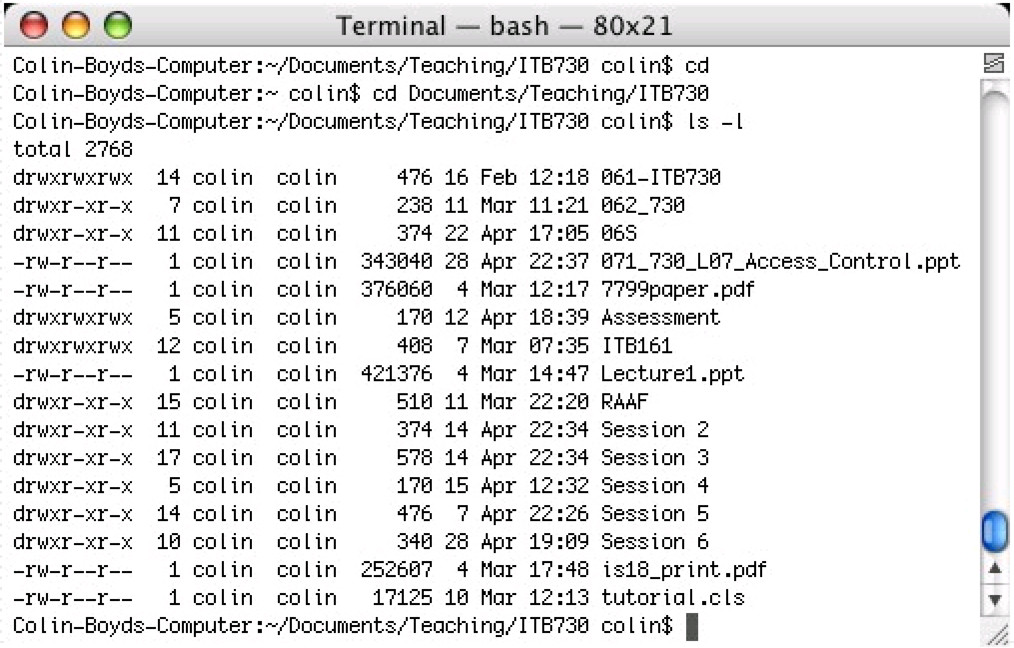
- Access Control Lists (ACL)
  - Attached to an object
  - Provides an access rule for a list of subjects
  - Simple means of enforcing policy
  - Does not scale well
- ACLs can be combined into an access control matrix covering access rules for a set of objects

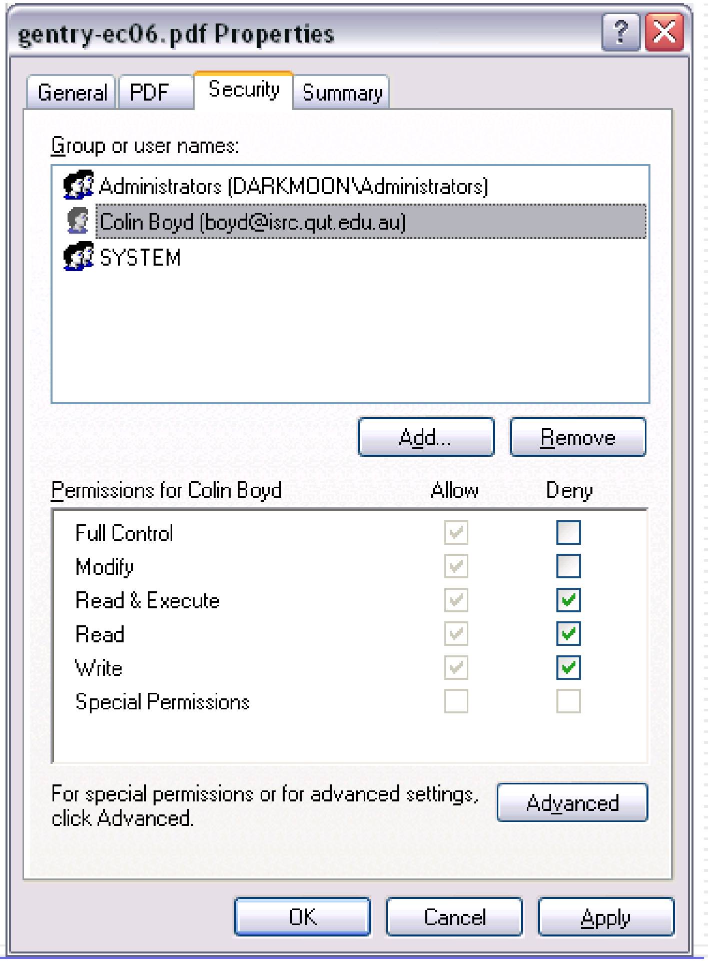
		Objects			
		O1	O2	O3	O4
Subjects	S1	rw	-	x	r
	S2	r	-	r	rw
	S3	-	x	-	-
	S4	rw	x	x	x

DAC in popular operating systems

Windows XP →

Apple OS X ↓



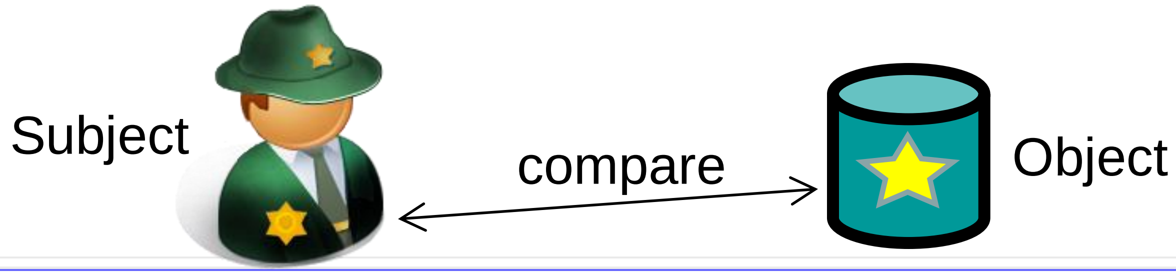


Basic concepts: MAC

- Mandatory access control
  - A central authority assigns access privileges
  - Usually implemented with security labels
    - Example: Clearance and classification levels
  - A system-wide **set of rules** is formed relating the attributes of the objects and subjects to the modes of access that are permitted
  - MAC is mandatory in the sense that the system is denying users full control over the access to the resources they create.
  - (SE)Linux includes MAC

Basic concepts: Labels

- Security Labels can be assigned to subjects and objects
  - Represents a specific security level, e.g. “Confidential” or “Secret”
- Object labels are assigned according to sensitivity
- Subject labels are determined by the authorization policy
- Access control decisions are made by comparing the subject label with the object label according to rules
- The set of decision rules is a security model
  - Used e.g. in the Bell-LaPadula and Biba models (see later)



Basic concepts: Combined MAC & DAC

- Combining access control approaches:
  - A combination of mandatory and discretionary access control approaches is often used
    - Mandatory access control is applied first:
    - If access is granted by the mandatory access control rules,
      - then the discretionary system is invoked
    - Access granted only if both approaches permit
  - This combination ensures that
    - no owner can make sensitive information available to unauthorized users, and
    - ‘need to know’ can be applied to limit access that would otherwise be granted under mandatory rules

Basic concepts: RBAC

- Role based access control
  - Access rights are based on the role of the subject, rather than the identity
    - A role is a collection of procedures or jobs that the subject performs
    - Examples: user, administrator, student, etc
    - A subject could have more than one role, and more than one subject could have the same role
  - RBAC can be combined with DAC and MAC

Security Models Introduction

- In order to describe an access policy, it is necessary to describe the entities that the access policy applies to and the rules that govern their behaviour.
- A security model provides this type of description.
- Security models have been developed to describe access policies concerned with:
  - Confidentiality (Bell-LaPadula,Clark-Wilson,Brewer-Nash,RBAC)
  - Integrity (Biba, Clark-Wilson, RBAC)
  - Prevent conflict of interest (Brewer-Nash, RBAC)

The Bell-LaPadula Model

Important Point:

The Bell-LaPadula model has its origins in the military’s need to maintain the confidentiality of classified information.

Bell-LaPadula Model: Overview

- While working for the Mitre Corporation in the 1970s, David E. Bell and Leonard J. LaPadula developed the Bell-La Padula Model in response to US Air Force concerns over the security of time-sharing mainframe systems.
- The Bell-LaPadula model focuses on the confidentiality of classified information – a Confidentiality-focussed Security Policy.
- The model is a formal state transition model of computer security policy that describes a set of access control rules enforced through the use of security labels on objects and clearances for subjects.

Bell-LaPadula Model: Information Flow

- Subjects are active entities in the system (for example users, processes, other computers), that cause information to flow among objects or change the system state.
- The Bell-LaPadula model is often called an *information flow model*. It is concerned with how information of different security sensitivity is allowed to flow amongst different objects

Bell-LaPadula Model: Hierarchical Security Levels

Top Secret
|
Secret
|
Confidential
|
Unclassified

- Security levels are typically used in military and national security domains
- Provide coarse-grained access control

Bell-LaPadula Model: Limitation of security levels

- Simple hierarchical levels alone are sometimes too coarse to implement adequate access control.
- A person (subject) with Secret clearance may not need access to <u>all</u> information files (objects) classified as Secret in order to perform their job.
- One of the principles of good security is to enforce access control based on 'need to know'.

Access Categories

- To implement the 'need to know' principle, define a set of non-ordered categories.
  - Subject and objects can have a set of categories in addition to their hierarchical security level;
- Example categories could be
  - Names of departments, such as:

Development – Production – Marketing – HR

- Not originally part of the Bell-LaPadula model

Bell-LaPadula Model: Security Labels

- Each subject and object has a Security Label
  - Subjects have a Maximum Security Label L^{SM}.
  - Subjects can use a Current Security Label $L^{SC} \leq L^{SM}$
  - Objects have a fixed Security Label L^{O}.
- The aim is to prevent subjects from accessing an object with a security label that is incompatible with the subject's security label.
- Subjects can chose to use a lower "current" label than their maximum label when accessing objects.

Bell La Padula Model:
Security Labels and Domination

- Security labels that are assigned to subjects and objects can consist of two components
  - a hierarchical level, and
  - a set of categories (not originally part of Bell-LaPadula)
- Label dominance
  - Let label L_A = (h-level$_A$, category-set$_A$)
  - Let label L_B = (h-level$_B$, category-set$_B$).
  - Then Label$_A$ dominates Label$_B$ iff
    - h-level$_B$ is less than or equal to h-level$_A$ and
    - category-set$_B$ is a subset of category-set$_A$.

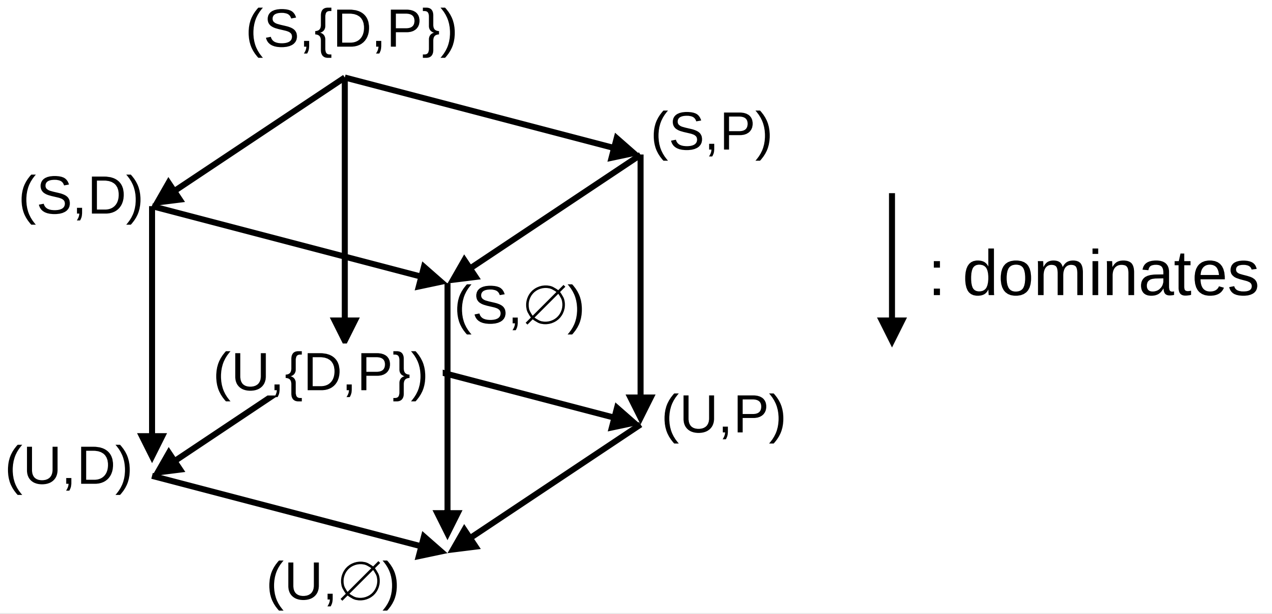
Partial Ordering of Labels

- Example: Define a label L = (h, c) where

  h $\in$ hierarchical set H = {Unclassified, Secret} = {U, S}

  c $\subseteq$ category set C = {Development, Production} = {D, P}

Partial ordering lattice



Definition of label dominance

- Labels defined as: $L = (h, c)$, $h \in H$ and $c \subseteq C$

  H: set of hierarchical levels, C: set of categories

- Example labels: $L_A = (h_A, c_A)$, $L_B = (h_B, c_B)$,
- Dominance: $L_A \geq L_B$ iff $(h_B \leq h_A) \wedge (c_B \subseteq c_A)$
  - In case $L_A = L_B$ then also $L_A \geq L_B$ and $L_B \geq L_A$
- Non-dominance cases: $L_A \not\geq L_B$
  - $(h_B > h_A) \wedge (c_B \subseteq c_A)$; insufficient security level
  - $(h_B \leq h_A) \wedge (c_B \not\subset c_A)$; insufficient category set
  - $(h_B > h_A) \wedge (c_B \not\subset c_A)$; insufficient level and category

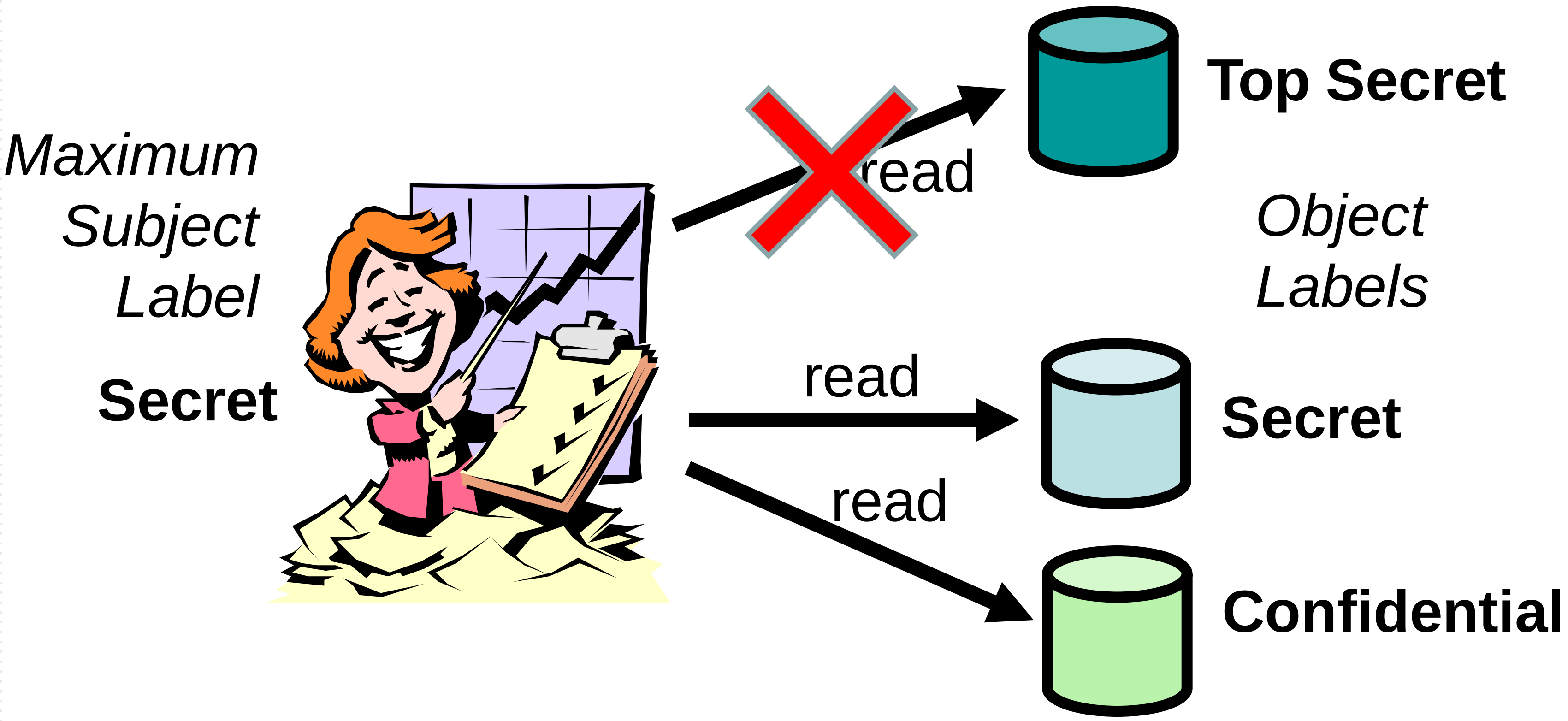
Bell-LaPadula Model: Security Properties

- In each state of a system the Bell-LaPadula model maintains three security properties:
  - ss-property (simple security)
  - * -property (star)
  - ds-property (discretionary security)

Bell-LaPadula Model: SS-Property: No Read Up

- Regulates read access
- The ss-property is satisfied if,
  - Subject Maximum Label L^{SM} dominates Object Label L^{O} for all current accesses where Subject has observe (read) access to Object:
- You can read a file if its hierarchical security level is lower than or equal to yours, and the category of this file is in your 'need to know' set.
- Traditionally known as the "no read-up" policy.
- In practice L^{SC} = max L^{O} of current accessed obj

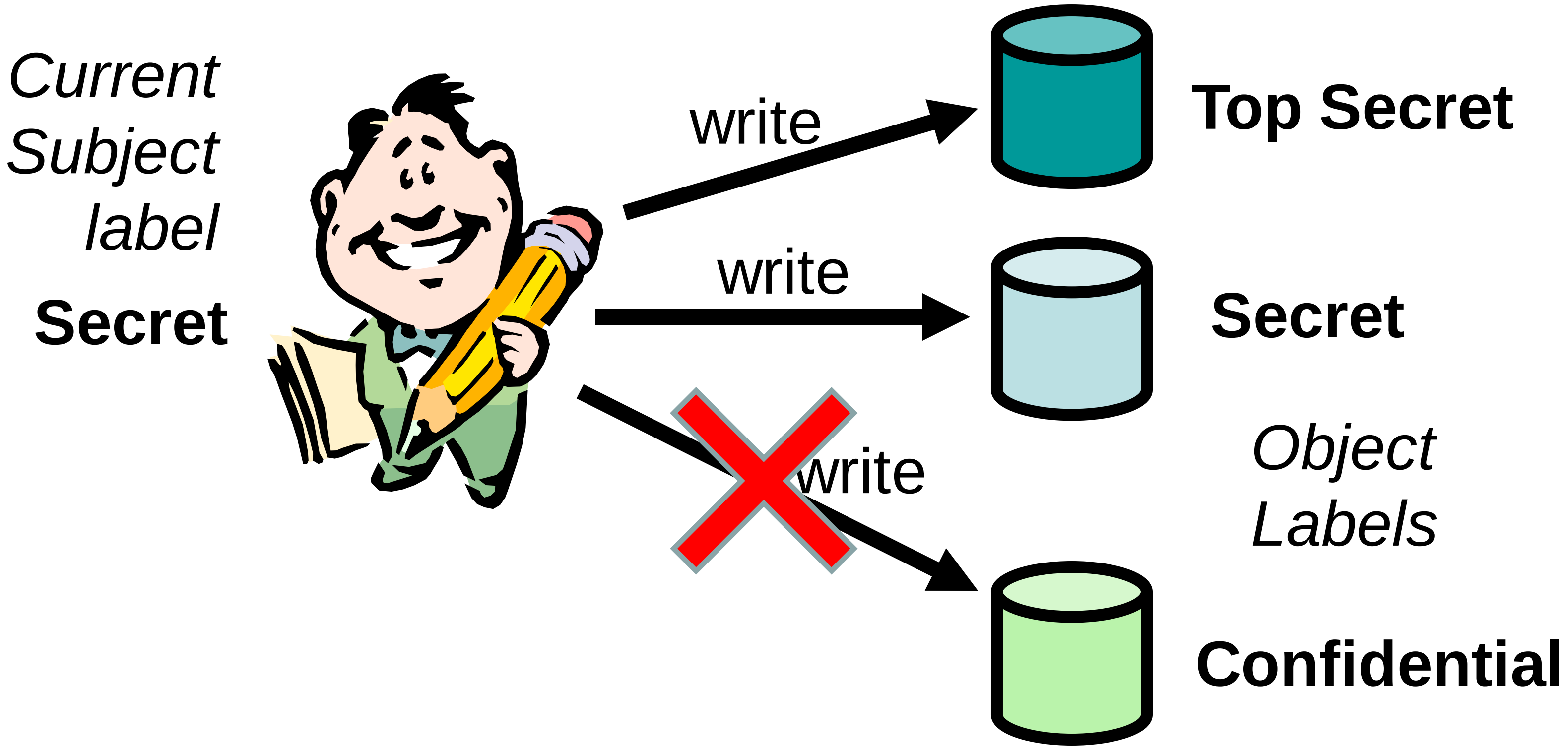
Bell-LaPadula Model:
SS-Property: No Read Up



Bell-LaPadula Model: *-Property: No Write Down

- Subjects working on information/tasks at a given label should not be allowed to write to a lower level because this could leak sensitive information.
- For example, you should only be able write to files with the same label as your label, or
- you could also write to files with a higher label than your label, but you should not be able to read those files.

Bell-LaPadula Model: *-Property: No Write Down

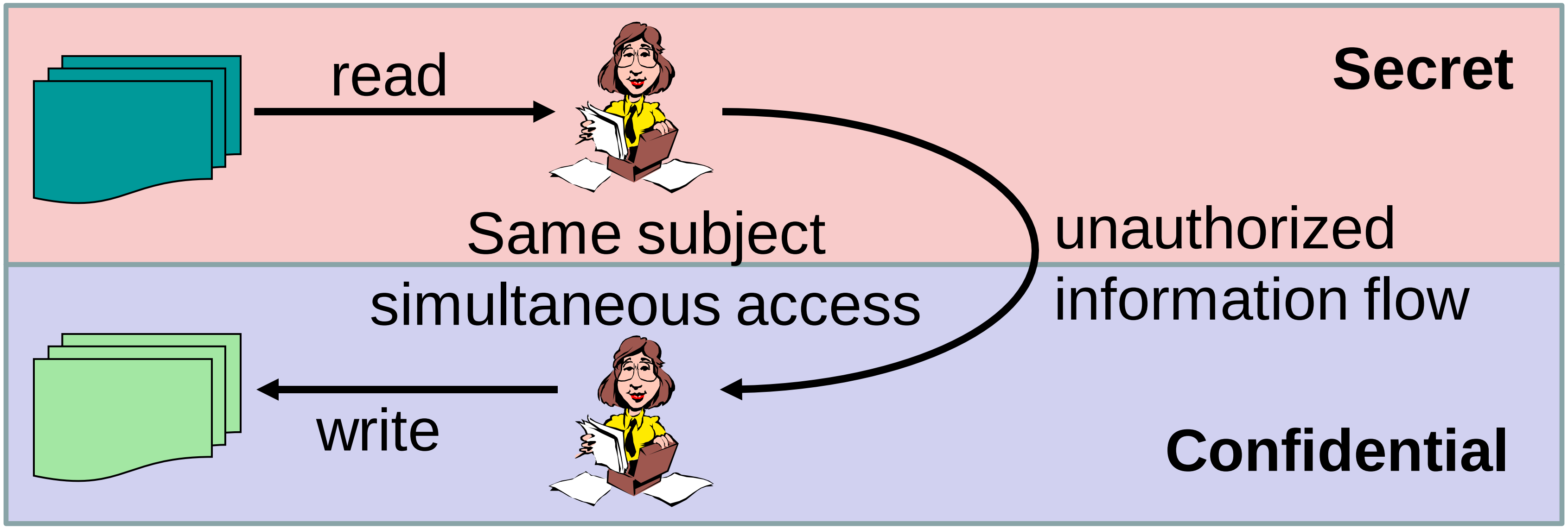


Bell-LaPadula Model: *-Property: Simultaneous read/write access

- The ss-property alone is not sufficient to prevent unauthorized information flow.
- A subject could chose read access to a high-level security object and chose write access to a low-level security object.
- This would enable data from a high-level object to be accessible to a subject with a low Maximum Security Label.
- Therefore, we also have to control simultaneous read-write accesses.

Subjects as (illegal) information channels

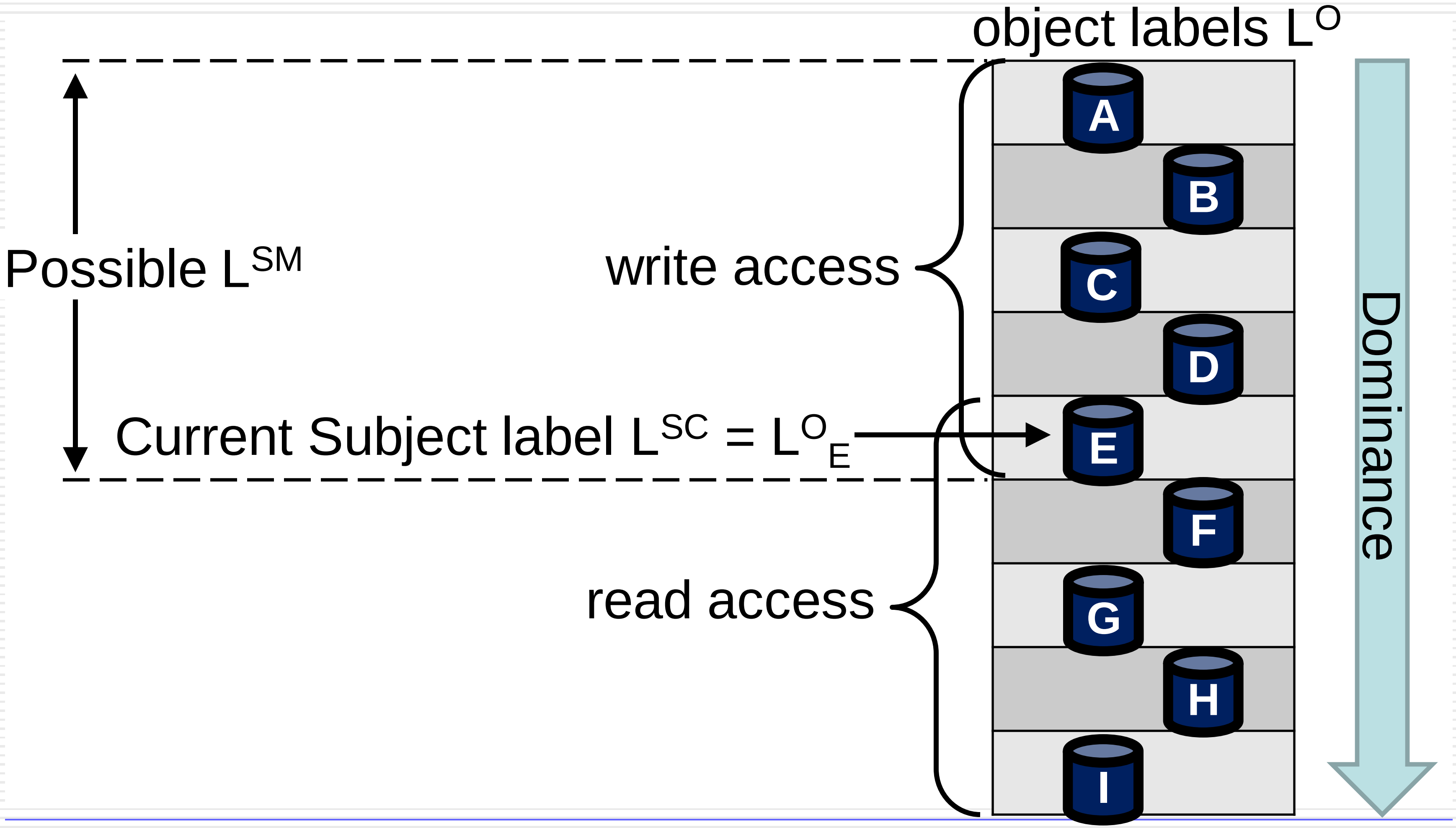
- Subjects could request write access to resources at low level while they have read access at high level
- Could cause information leakage:



Bell-LaPadula Model: *-Property: No Write Down

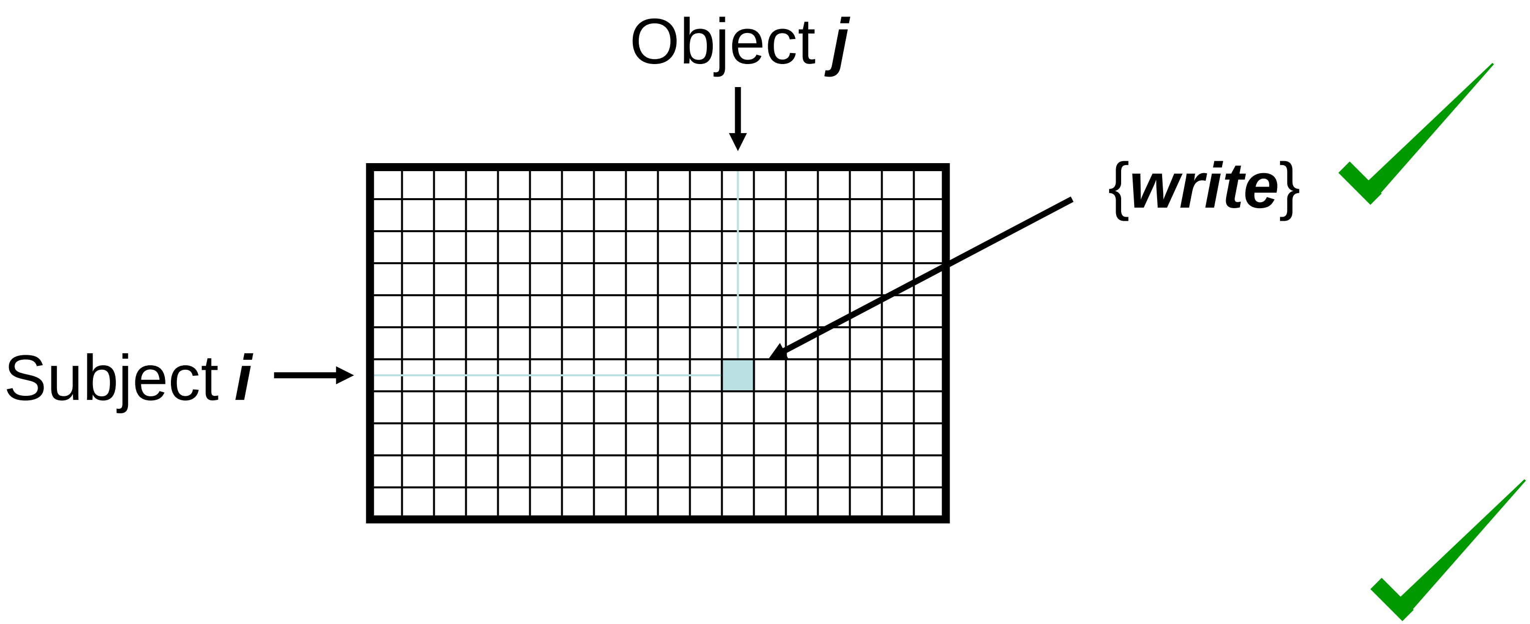
- The *-property (star property) regulates simultaneous read and write access.
- The *-property is satisfied if,
  - For all cases where a Subject has simultaneous alter (write or append) access to $Object_A$ and observe (read) access to $Object_B$, then the security label of $Object_A$ dominates the security label of $Object_B$
- This is known as the “no write-down” policy
- In practice $L^{O}(w) \geq L^{SC}$ (every object accessed for writing must dominate the current subject label)

Bell-LaPadula label relationships



Bell-LaPadula Model:
DS Property: Matrix Entry

- M(i,j) satisfies current access request



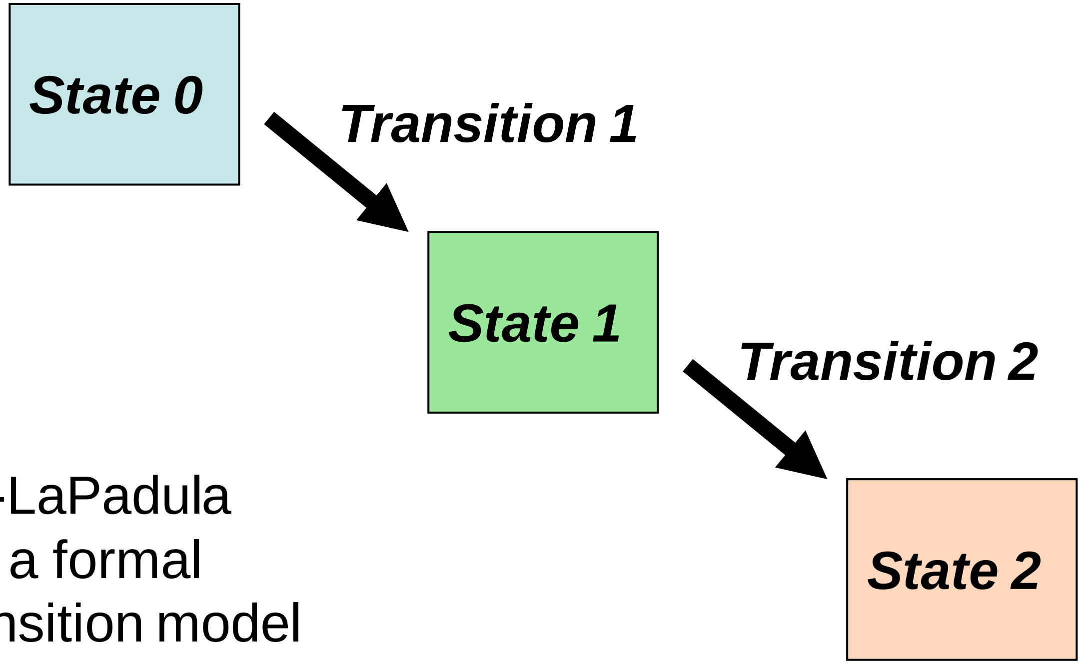
? Access request : {subject = ***i***, object = ***j***, access = ***write***}

Bell-LaPadula Model: DS Property: Matrix Entry

- The ds-property (discretionary security property) is a Bell-LaPadula security model rule that demands that the current access by subject S to object O is permitted by the current access permission matrix M.
- This was the original method to enforce need-to-know in Bell-LaPadula.

Bell-LaPadula Model: Basic Security Theorem

- If the initial state is secure and all state transitions in a system are secure, then all subsequent states will also be secure no matter what inputs occur.



The Bell-LaPadula model is a formal state transition model

Bell-LaPadula Tranquility

- Bell-LaPAdula does not specify rules for changing access control policies (i.e. changing labels on subjects and objects).
  - assumes tranquility: access control policies do not change.
- Operational model: users get clearances and objects are classified following given rules.
- The system is set up to enforce MLS (Multi-Level Security) policies for the given clearances and classifications.
- Changes to clearances and classifications requires external input.

The Biba Model for Integrity

- In Biba, subjects and objects have integrity labels
- *The Biba Simple Integrity Axiom* states that a subject at a given level of integrity must not read an object at a lower integrity level (**no read down**).
- *The Biba * (star) Integrity Axiom* states that a subject at a given level of integrity must not write to any object at a higher level of integrity (**no write up**).
- Opposite to Bell-LaPadula
- Combining Biba and Bell-LaPadula results in a model where subjects can only read and write at their own level

The Brewer-Nash Chinese Wall Model

Brewer-Nash model: Overview

- The Brewer-Nash model is a confidentiality model for the commercial world.
  - In a consultancy-based business, analysts have to ensure that no conflicts of interest arise in respect to dealings with different clients.
  - A conflict of interest is a situation where someone in a position of trust has competing professional and/or personal interests and their ability to carry out their duties and responsibilities objectively is compromised or may be seen to be compromised.

- Rule: There must be no information flow that causes a ‘conflict of interest’.

Brewer-Nash model: Sanitized and Unsanitized Information

- Assume that a consultancy-based business has confidential information pertaining to individual companies that are its clients.
  - Information that can be identified as belonging to a particular company is deemed to be unsanitized.
  - Information that cannot be identified as belonging to a particular company is deemed to be sanitized.
  - Also, where information is held regarding a company but it is not confidential (already public knowledge say), this information is not subject to the policy implemented by this model.
- The Brewer-Nash model is concerned with the flow of unsanitized information.
  - Sanitized information flow is not of concern in this model.

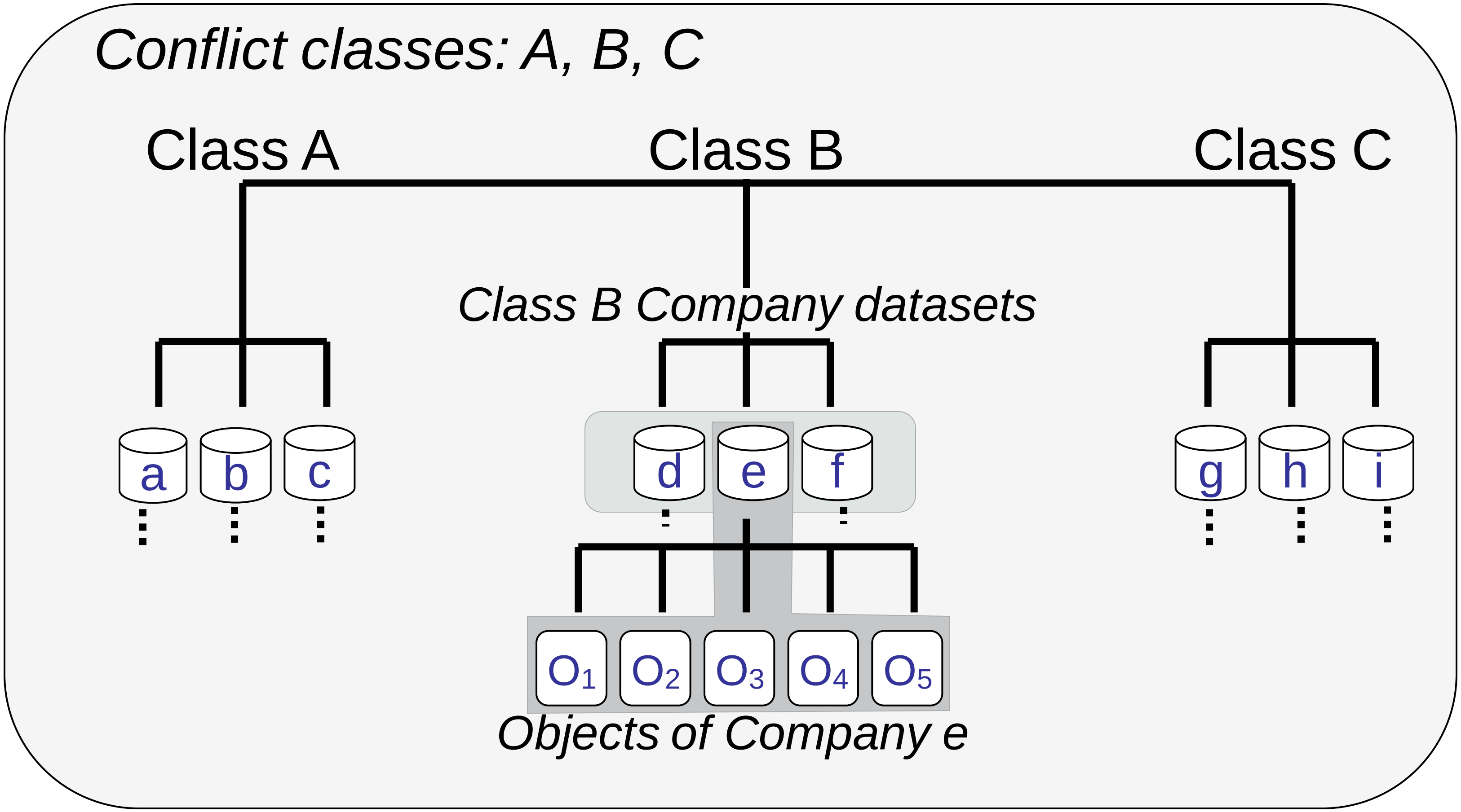
Brewer-Nash model: Objects, Datasets & Conflict Classes

- Objects:
  - Individual items of information belonging to a single corporation are stored as objects;
  - Each object has a security label
  - Security labels contain information about which company the object belongs to, and the ‘conflict of interest’ class the object belongs to.
- Company datasets:
  - All objects which concern the same corporation are grouped together into a company dataset;
- Conflict classes:
  - All company datasets whose corporations are in competition are grouped together into the same conflict of interest class.

Brewer-Nash model: Chinese Wall Example

- Scenario:
  - A marketing agency handles accounts for companies involved in:
    - confectionary manufacture (Class A),
    - car rental (Class B), and
    - clothing (Class C).
  - The car rental companies are:
    - Hurts (company d),
    - Aviz (company e), and
    - Eurocar (company f).
- Let's say a marketing analyst has previously only accessed object O_2 in the company dataset e (Aviz) of conflict class B (car rental)

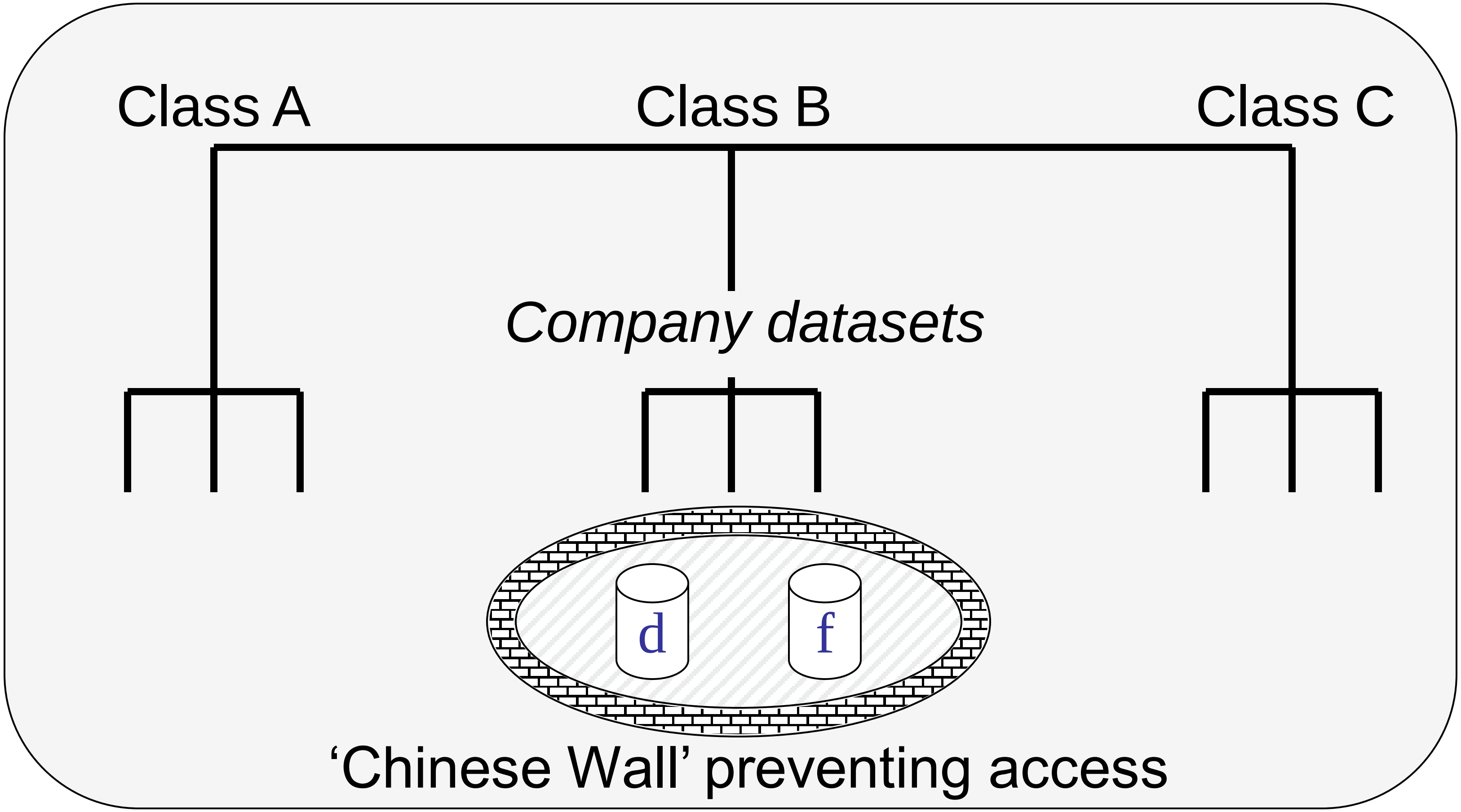
Brewer-Nash model: Chinese Wall Example



Brewer-Nash model: Chinese Wall Example

- The analyst cannot now access any objects in company datasets d or f (Hurts or Eurocar).
  - Aviz, Hurts and Eurocar are in competition with each other. Accessing information belonging to d or f would lead to a conflict of interest.
- The analyst can access an object in conflict class A (confectionary) or conflict class C (clothing).
  - Insider information about confectionary and clothing companies does not represent a conflict of interest with Aviz (car rental) as confectionary and clothing companies are not in direct competition with car rental companies.

Brewer-Nash model: Mandatory Access Control



Brewer-Nash model: Mandatory Access Rules

- Initially the analyst may access any file from any dataset
  - In our example, the analyst can access object 2, Company e dataset, Conflict class B.
- Thereafter, the analyst:
  - can access any other object (file) in Dataset e, or
  - can access any object in the hierarchy of Conflict classes A or C,
  - but cannot access any object in Dataset d or f of Conflict class B.

Brewer-Nash model: Access Matrix (N)

Object / Subject	O_1	O_2	O_3	O_4	…	O_j	…
S_1	F	T	F	F		F	
⋮							
S_i	F	F	T	F		T	
⋮							

Brewer-Nash model: Access Matrix (N)

- The Access Matrix N determines a subject's right to access an object.
  - The rows of N represent subjects and the columns represent objects.
  - The elements in N are boolean values -- true or false.
  - Element N(i,j) indicates whether subject i has been granted previous access to object j.
  - Initially all entries of the matrix N are set to f (false) – no objects have been previously accessed by any subjects.
  - When subject i is granted access to object j, N(i,j) is set to t (true).
- In order to determine if an access request can be approved, all previous accesses that have occurred must be considered.

Brewer-Nash model: Simple Security (ss) Property

- Access to object O_j is granted to subject S_i only if O_j belongs to:
  - A company dataset CD already accessed by the subject (i.e. O_j is in CD and $N(i,k) = t$ for some O_k in CD)

  or

  - An entirely different conflict of interest class COI

  (i.e. O_j is in COI and for all objects O_k in COI, $N(i,k) = f$)

Brewer-Nash model: Star (*) Security Property

- Suppose two analysts, user A and user B, have the following access:
  - User A has access to information about car rental company **e** and confectionary company **a**.
  - User B has access to information about car rental company **d** and confectionary company **a**.
- If user A reads information from company **e** and writes it to a company **a** object, then user B has access to company **e** information.
- This should not be permitted because of the conflict of interest between company **e** and company **d**.

Brewer-Nash model: Star (*) Security Property

- Write access is only permitted if:
  - access is permitted by the **ss** rule, and
  - no object can be read which is in a different company dataset from the one for which write access is requested and contains unsanitized information.

- In other words, write access is granted only if no other object (with unsanitized data) can be currently read which is in a different company dataset (in any conflict class)

The Clark-Wilson Model

Clark Wilson model: Overview

- The Clark-Wilson Security model is an *integrity* model for the commercial environment.
- There is an emphasis on controlling transaction processing.
- The Clark-Wilson Security model provides a formal model for commercial integrity
  - The model attempts to prevent unauthorised modification of data, fraud and errors.

Clark Wilson model: Overview

- The Clark-Wilson Security model attempts to follow the conventional controls used in bookkeeping and auditing through certification and enforcement.
- Data is divided into two types
  - Unconstrained data items (UDI)
  - Constrained data items (CDI)
- CDIs cannot be accessed directly by users - they must be accessed through a transformation procecure (TP)
- In certain circumstances UDI may become CDI

Clark-Wilson model: System Integrity

- Internal consistency:
  - Is the internal state of the system consistent at all times?
  - This can be enforced by integrity verification procedures (IVPs)
  - The IVPs certify that the CDIs are in a valid state
  - The TPs must preserve state validity

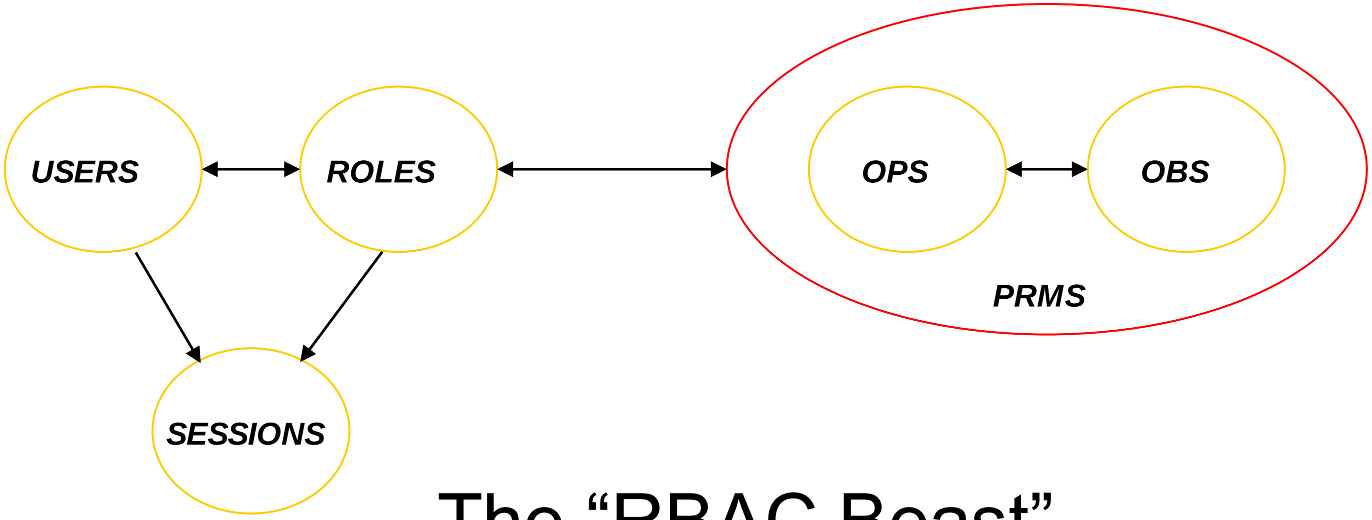
Clark Wilson model: Security Requirements Overview

- Every user must be identified and authenticated.
- Each data item can only be manipulated by a particular set of allowed programs.
- Each user can run only a particular set of programs.
- Separation of duty and well-formed transaction rules must be enforced by the system.
- Auditing log must be maintained.

The RBAC Model
Role Based Access Control

Role-Based Access Control

- A brief introduction
  - Based on Proposed NIST Standard for Role-Based Access Control
  - http://csrc.nist.gov/rbac/sandhu-ferraiolo-kuhn-00.pdf

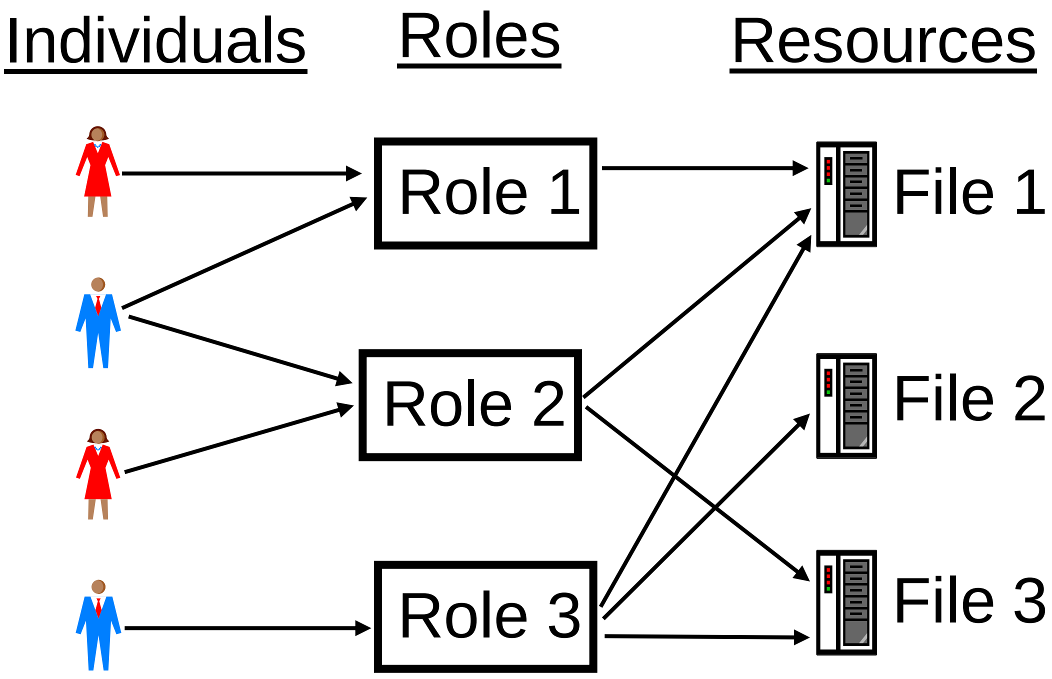


The "RBAC Beast"

RBAC rationale

- A user has access to an object based on the assigned role.
- Roles are defined based on job functions.
- Permissions are defined based on job authority and responsibilities within a job function.
- Operations on an object are invocated based on the permissions.
- The object is concerned with the user's role and not the user.

RBAC Flexibility



User's change frequently, roles don't

- RBAC can be configured to do MAC
- RBAC can be configured to do DAC

RBAC Privilege Principles

- Roles are engineered based on the principle of least privilege .
- A role contains the minimum amount of permissions to instantiate an object.
- A user is assigned to a role that allows him or her to perform only what's required for that role.
- No single role is given more permission than the same role for another user.

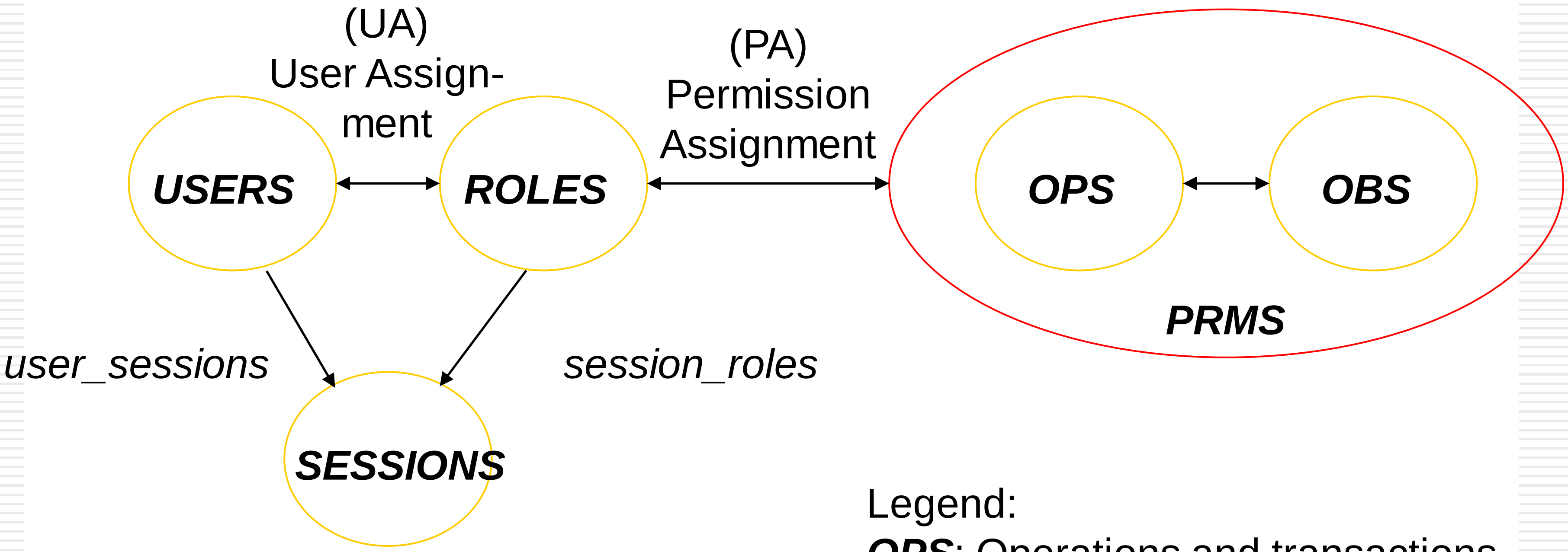
RBAC Framework

- Core Components
- Constraining Components
  - Hierarchical RBAC
    - Allows roles to be defined in a hierarchy, and role inheritance
  - Constrained RBAC
    - Can prevent conflict of interest in two ways
    - SSD (Static Separation of Duties) prevents assignment of conflicting roles
    - DSD (Dynamic Separation of Duties) allows assignment of conflicting roles, but prevents their simultaneous invocation

RBAC Core Components

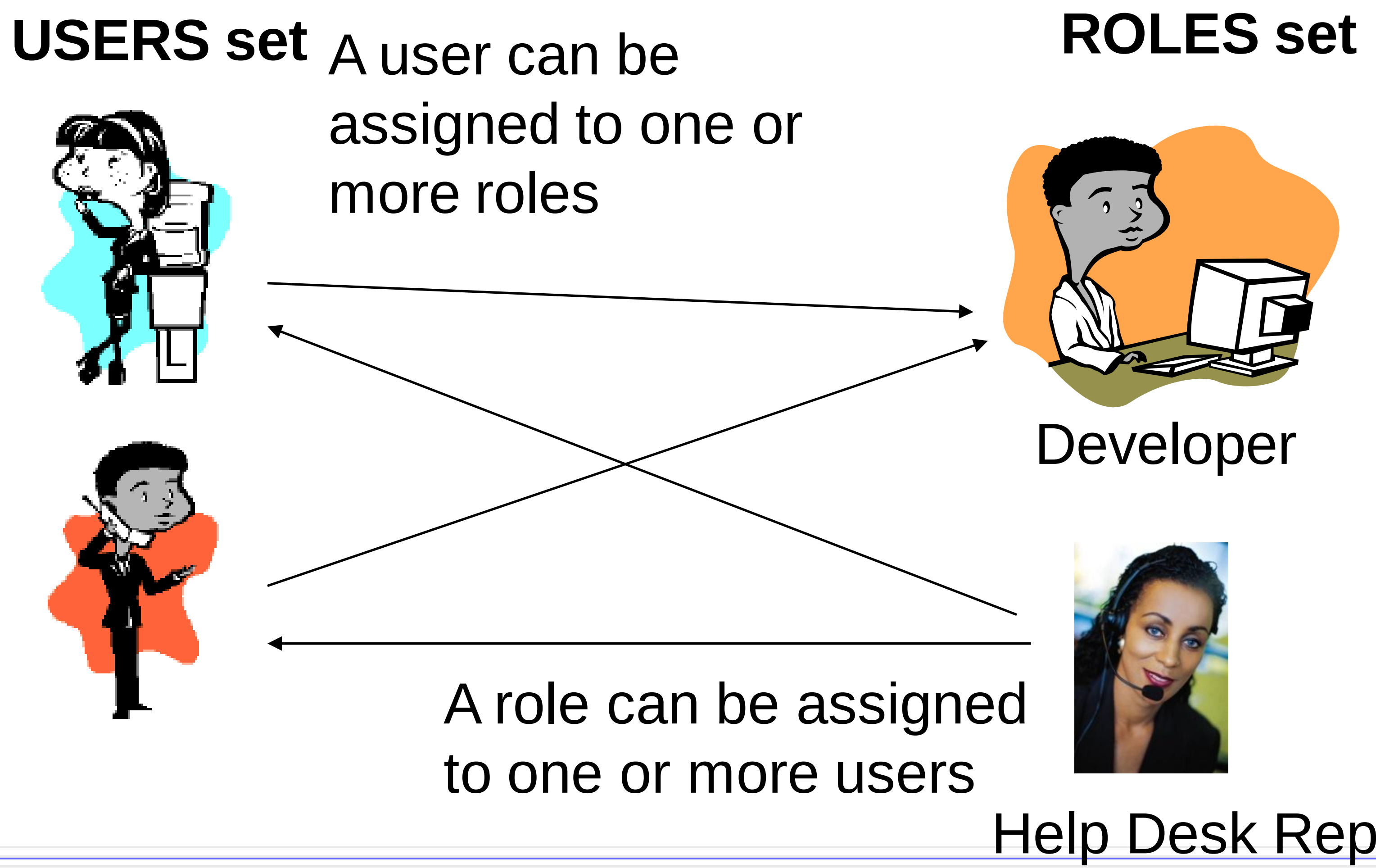
- Defines:
  - USERS
  - ROLES
  - OPERATIONS (*ops*)
  - OBJECTS (*obs*)
  - User Assignments (*ua*)
    - assigned_users
  - Permissions (*prms*)
    - Assigned Permissions
    - Object Permissions
    - Operation Permissions
  - Sessions
    - User Sessions
    - Available Session Permissions
    - Session Roles

Core RBAC



Legend:
OPS: Operations and transactions
OBS: Objects, databases, files
PRMS: Permissions

RBAC UA (user assignment)



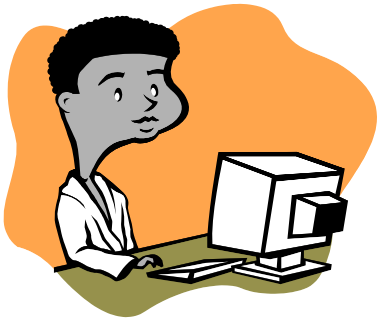
RBAC PA (prms assignment)

ROLES set

PRMS set

A prms can be assigned to one or more roles

Create
Delete
Drop



Admin.DB1

View
Update
Append

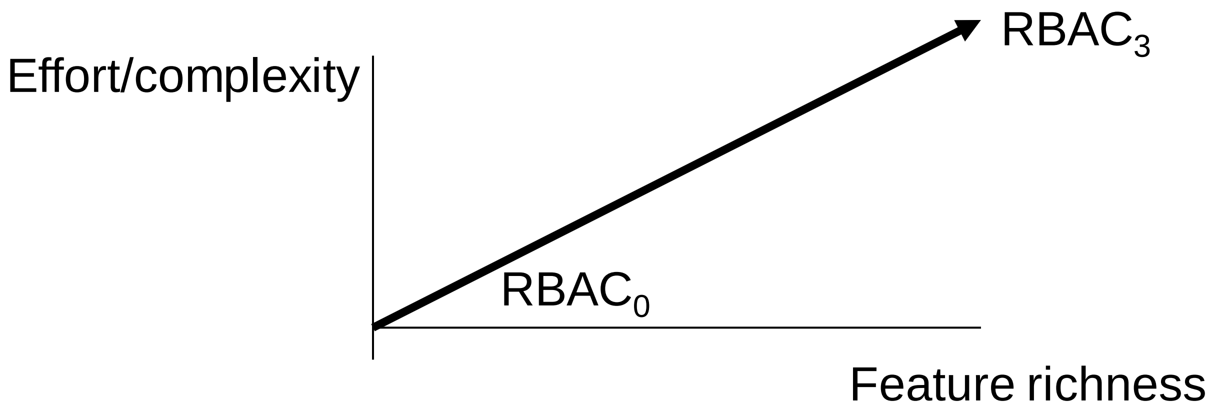
A role can be assigned to one or more prms

User.DB1

RBAC Models

Core RBAC

Models	Hierarchical	Constrained
$RBAC_0$	No	No
$RBAC_1$	Yes	No
$RBAC_2$	No	Yes
$RBAC_3$	Yes	Yes



RBAC Operational Aspects

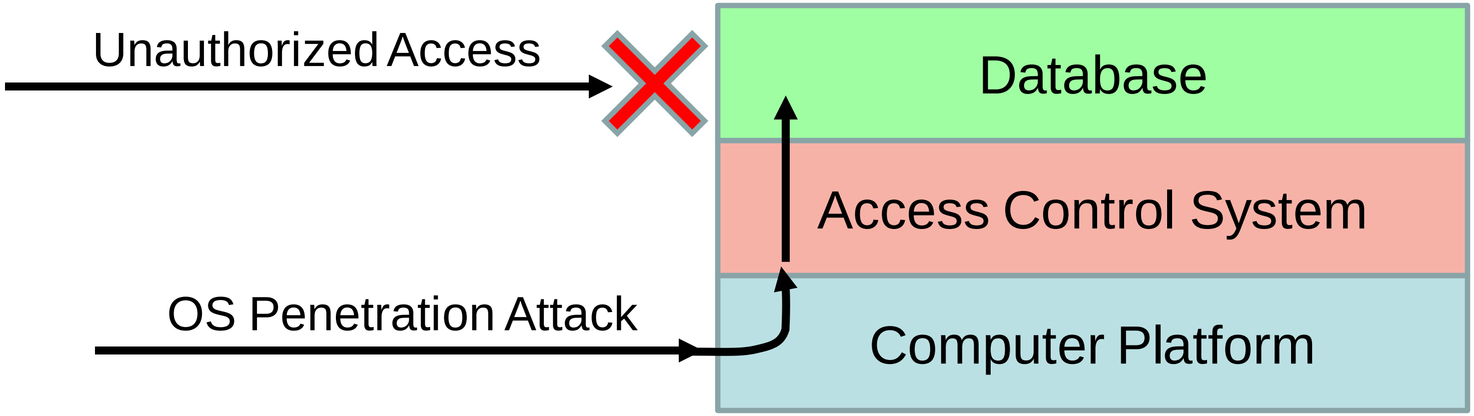
- System Level Functions
  - Creation of user sessions
  - Role activation/deactivation
  - Constraint enforcement
  - Access Decision Calculation
- Administrative Operations
  - Create, Delete, Maintain elements and relations
- Implementation challenge
  - Large number of different roles can become a problem in practical implementations

AC Limitations: Covert Channels

- Covert Channels are Communications channels that allow transfer of information in a manner that violates the system's security policy.
  - Storage channels: e.g. through operating system messages, file names, etc.
  - Timing channels: e.g. through monitoring system performance
- Orange Book: 100 bits per second is 'high' bandwidth for storage channels, no upper limit on timing channels.
- Security models do not consider covert channels

AC Limitations: Platform Security

- AC (Access Control) systems assume the integrity and security of the platform on which they are implemented.
- In case access to a database system is protected by AC, but the OS can not protect the AC functionality, then the AC System can be bypassed by attackers.



Review

- Physical or logical AC
- AC philosophy and basic concepts
- Authentication and AC sequence
  - Identification – Authentication – Access Control
- Authentication and Access phases
  - Registration/Authorization – Authentication/Access Control
- MAC, DAC, RBAC
- Formal Models
  - Bell-LaPadula, Biba, Brewer-Nash, Clark-Wilson, RBAC
- Covert channels and platform security