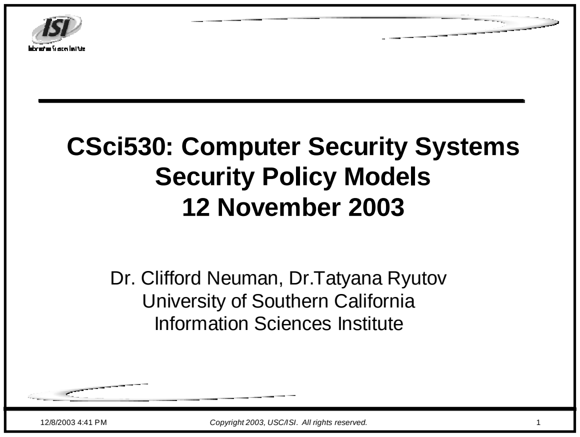
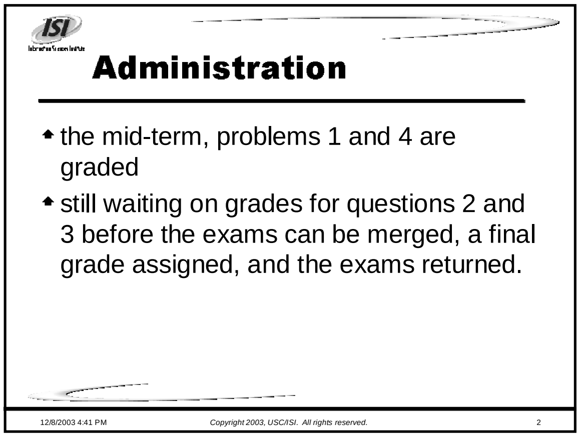


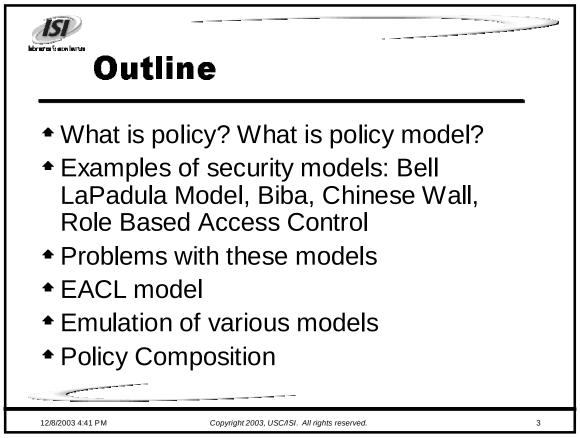
CSci530: Computer Security Systems
Security Policy Models
12 November 2003

Dr. Clifford Neuman, Dr.Tatyana Ryutov
University of Southern California
Information Sciences Institute



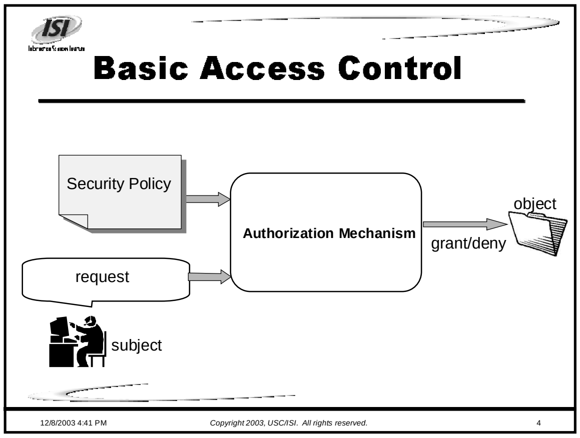
Administration

- the mid-term, problems 1 and 4 are graded
- still waiting on grades for questions 2 and 3 before the exams can be merged, a final grade assigned, and the exams returned.

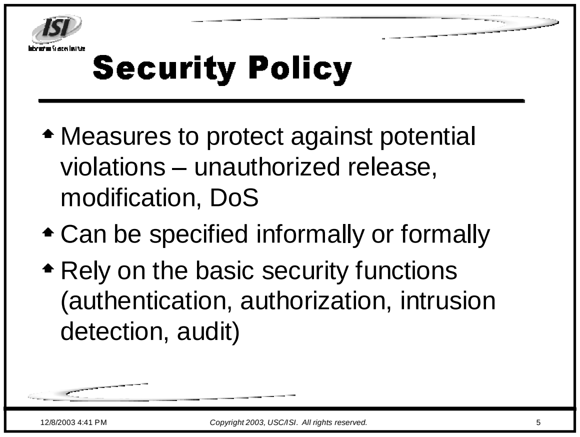


Outline

- What is policy? What is policy model?
- Examples of security models: Bell LaPadula Model, Biba, Chinese Wall, Role Based Access Control
- Problems with these models
- EACL model
- Emulation of various models
- Policy Composition

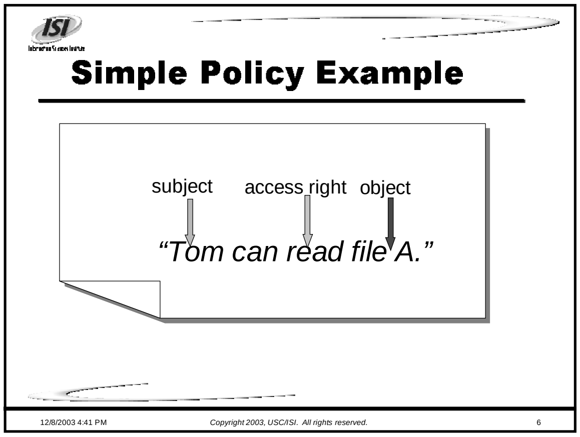


Basic Access Control

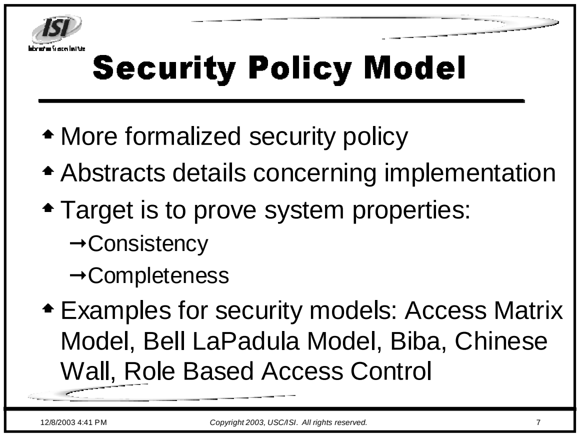


Security Policy

- Measures to protect against potential violations – unauthorized release, modification, DoS
- Can be specified informally or formally
- Rely on the basic security functions (authentication, authorization, intrusion detection, audit)

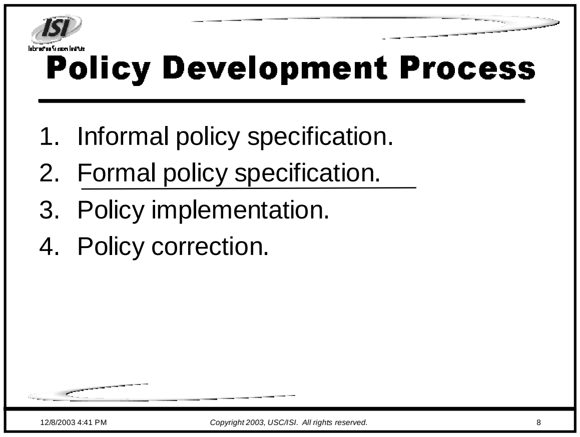


Simple Policy Example



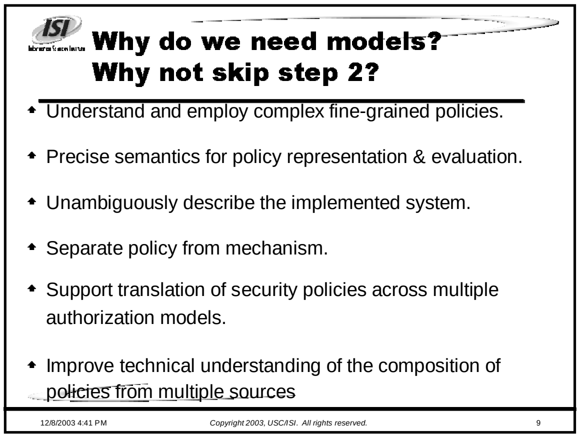
Security Policy Model

- More formalized security policy
- Abstracts details concerning implementation
- Target is to prove system properties:
  - →Consistency
  - →Completeness
- Examples for security models: Access Matrix Model, Bell LaPadula Model, Biba, Chinese Wall, Role Based Access Control



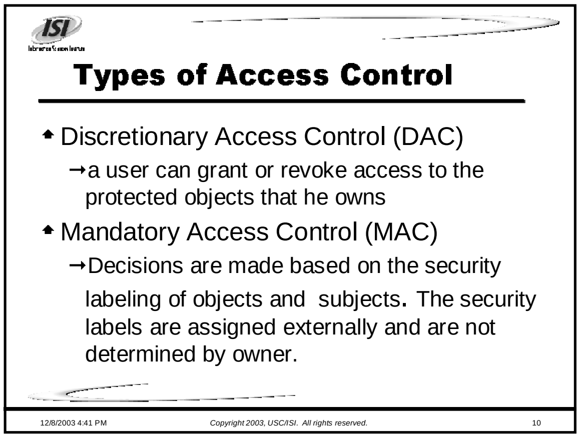
Policy Development Process

1. Informal policy specification.
2. Formal policy specification.
3. Policy implementation.
4. Policy correction.



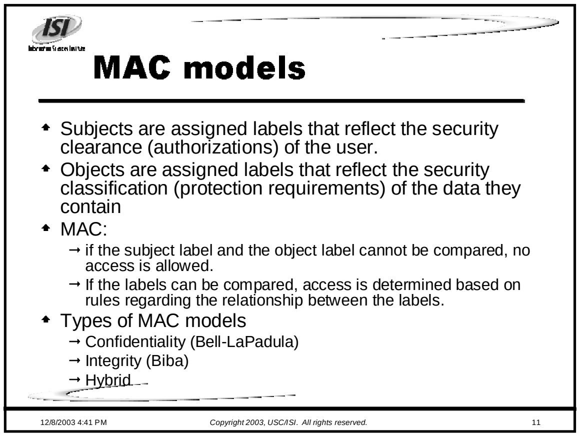
Why do we need models? Why not skip step 2?

- Understand and employ complex fine-grained policies.
- Precise semantics for policy representation & evaluation.
- Unambiguously describe the implemented system.
- Separate policy from mechanism.
- Support translation of security policies across multiple authorization models.
- Improve technical understanding of the composition of policies from multiple sources



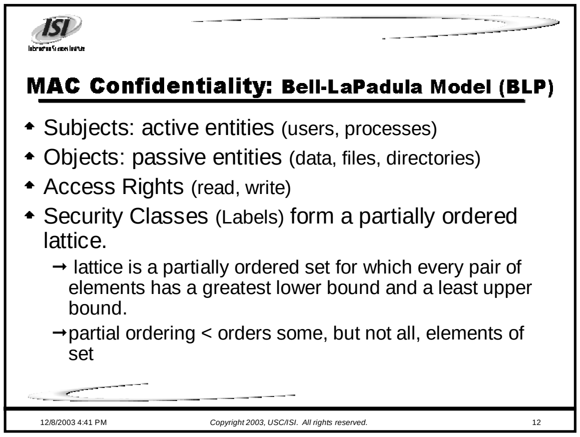
Types of Access Control

- Discretionary Access Control (DAC)
  - →a user can grant or revoke access to the protected objects that he owns
- Mandatory Access Control (MAC)
  - →Decisions are made based on the security labeling of objects and subjects**.** The security labels are assigned externally and are not determined by owner.



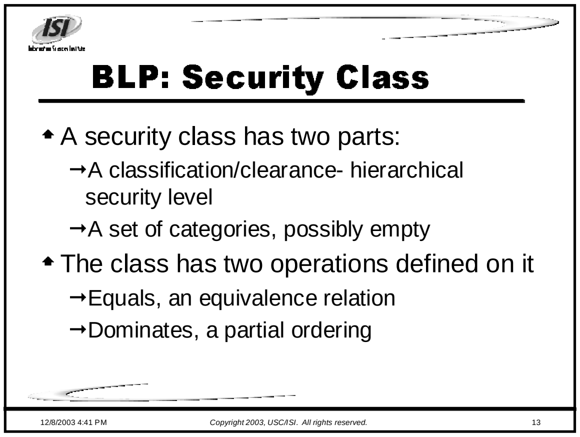
MAC models

- Subjects are assigned labels that reflect the security clearance (authorizations) of the user.
- Objects are assigned labels that reflect the security classification (protection requirements) of the data they contain
- MAC:
  - → if the subject label and the object label cannot be compared, no access is allowed.
  - → If the labels can be compared, access is determined based on rules regarding the relationship between the labels.
- Types of MAC models
  - → Confidentiality (Bell-LaPadula)
  - → Integrity (Biba)
  - → Hybrid



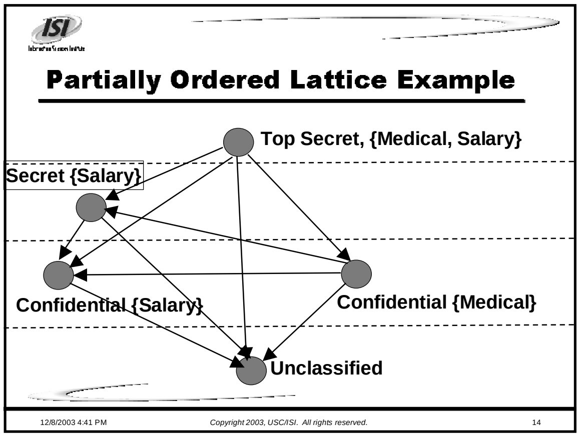
MAC Confidentiality: Bell-LaPadula Model (BLP)

- Subjects: active entities (users, processes)
- Objects: passive entities (data, files, directories)
- Access Rights (read, write)
- Security Classes (Labels) form a partially ordered lattice.
  - → lattice is a partially ordered set for which every pair of elements has a greatest lower bound and a least upper bound.
  - →partial ordering < orders some, but not all, elements of set

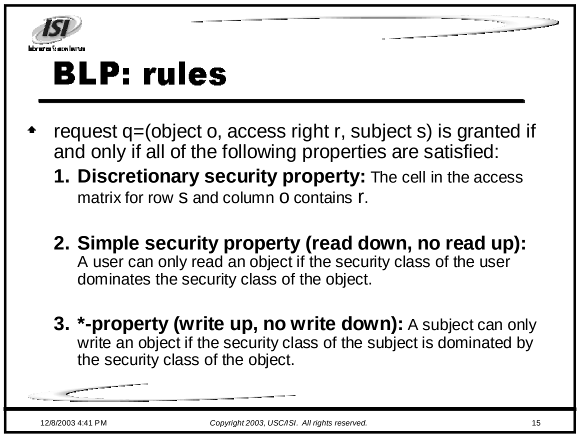


BLP: Security Class

- A security class has two parts:
  - → A classification/clearance- hierarchical security level
  - → A set of categories, possibly empty
- The class has two operations defined on it
  - → Equals, an equivalence relation
  - → Dominates, a partial ordering

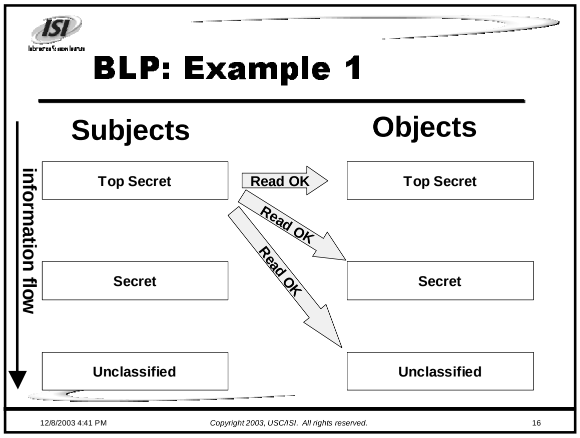


Partially Ordered Lattice Example

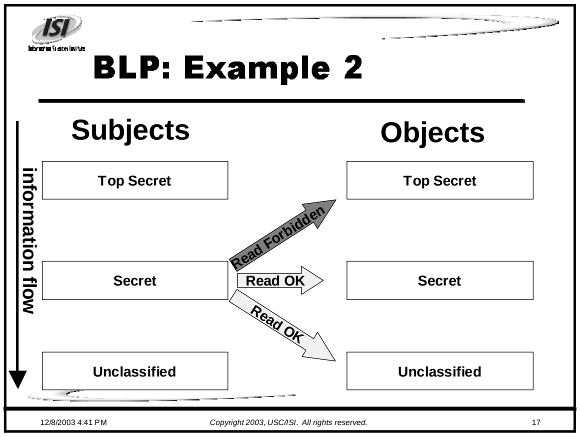


BLP: rules

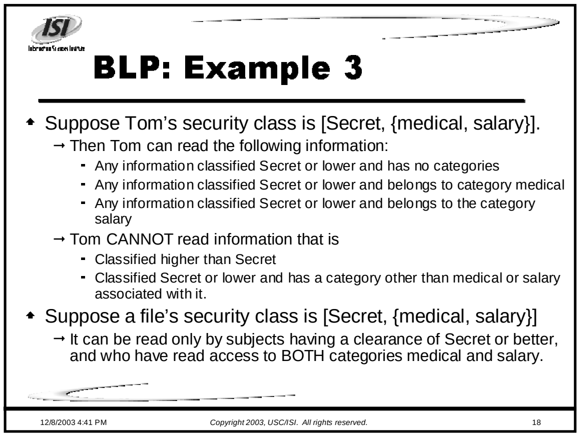
- request q=(object o, access right r, subject s) is granted if and only if all of the following properties are satisfied:
  1. **Discretionary security property:** The cell in the access matrix for row s and column o contains r.
  2. **Simple security property (read down, no read up):** A user can only read an object if the security class of the user dominates the security class of the object.
  3. ***-property (write up, no write down):** A subject can only write an object if the security class of the subject is dominated by the security class of the object.



BLP: Example 1

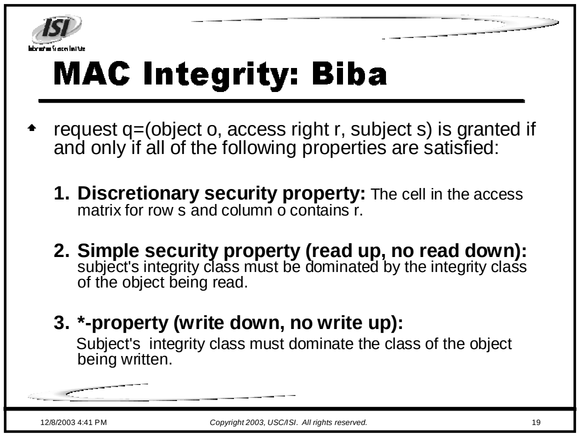


BLP: Example 2



BLP: Example 3

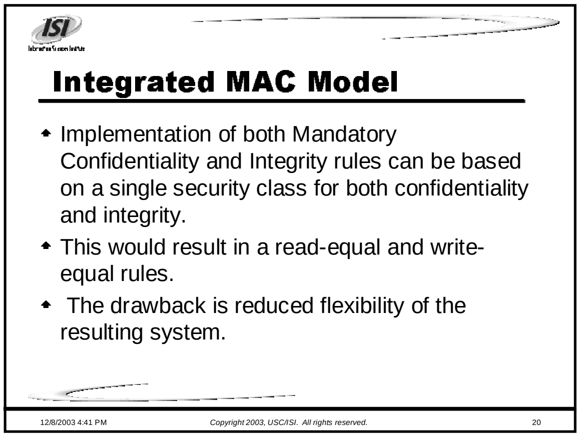
- Suppose Tom's security class is [Secret, {medical, salary}].
  - → Then Tom can read the following information:
    - Any information classified Secret or lower and has no categories
    - Any information classified Secret or lower and belongs to category medical
    - Any information classified Secret or lower and belongs to the category salary
  - → Tom CANNOT read information that is
    - Classified higher than Secret
    - Classified Secret or lower and has a category other than medical or salary associated with it.
- Suppose a file's security class is [Secret, {medical, salary}]
  - → It can be read only by subjects having a clearance of Secret or better, and who have read access to BOTH categories medical and salary.



MAC Integrity: Biba

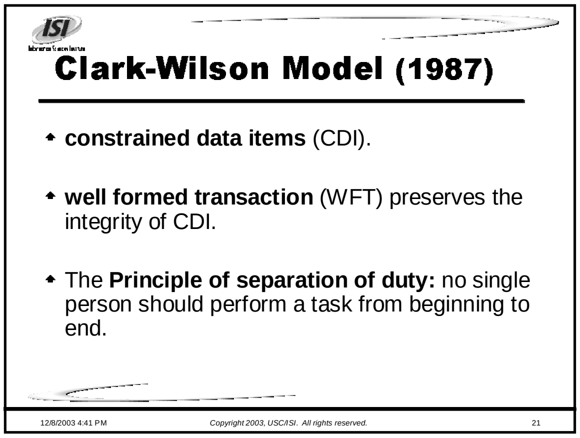
- request q=(object o, access right r, subject s) is granted if and only if all of the following properties are satisfied:

1. **Discretionary security property:** The cell in the access matrix for row s and column o contains r.
2. **Simple security property (read up, no read down):** subject's integrity class must be dominated by the integrity class of the object being read.
3. ***-property (write down, no write up):** Subject's integrity class must dominate the class of the object being written.



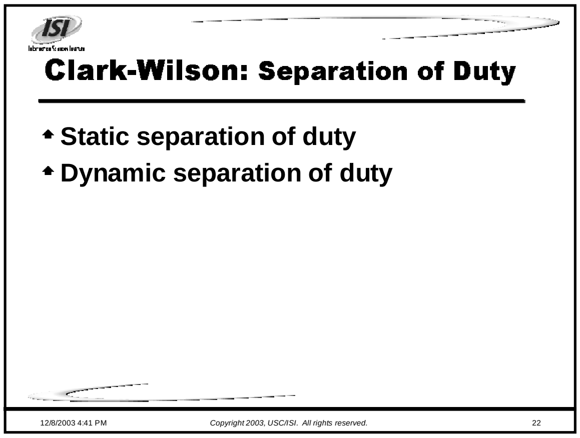
Integrated MAC Model

- Implementation of both Mandatory Confidentiality and Integrity rules can be based on a single security class for both confidentiality and integrity.
- This would result in a read-equal and write-equal rules.
- The drawback is reduced flexibility of the resulting system.



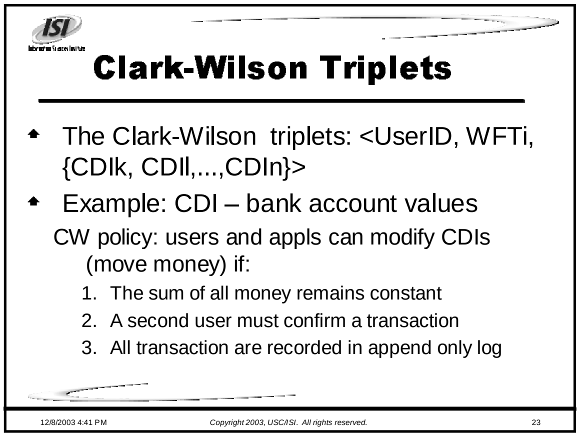
Clark-Wilson Model (1987)

- **constrained data items** (CDI).
- **well formed transaction** (WFT) preserves the integrity of CDI.
- The **Principle of separation of duty:** no single person should perform a task from beginning to end.



Clark-Wilson: Separation of Duty

- **Static separation of duty**
- **Dynamic separation of duty**

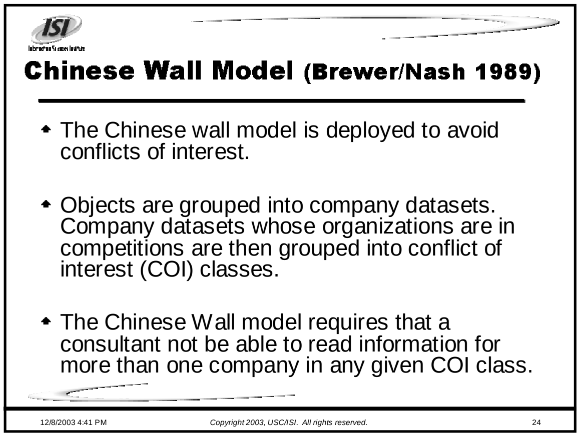


Clark-Wilson Triplets

- The Clark-Wilson triplets: <UserID, WFTi, {CDIk, CDIl,...,CDIn}>
- Example: CDI – bank account values

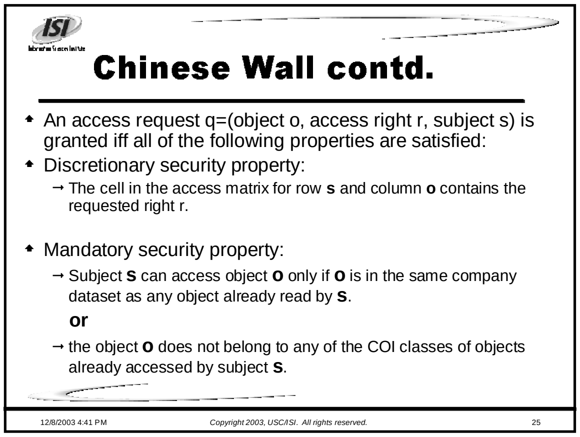
CW policy: users and appls can modify CDIs (move money) if:

1. The sum of all money remains constant
2. A second user must confirm a transaction
3. All transaction are recorded in append only log



Chinese Wall Model (Brewer/Nash 1989)

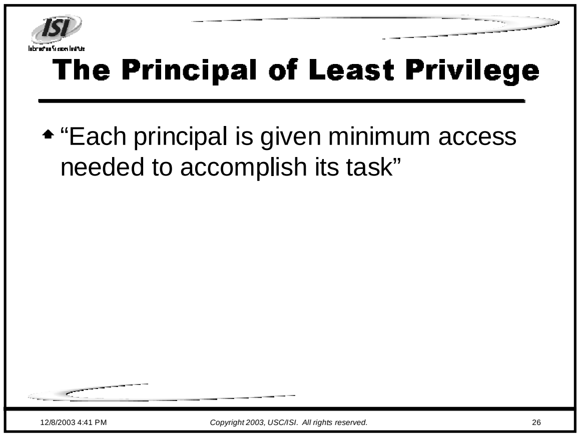
- The Chinese wall model is deployed to avoid conflicts of interest.
- Objects are grouped into company datasets. Company datasets whose organizations are in competitions are then grouped into conflict of interest (COI) classes.
- The Chinese Wall model requires that a consultant not be able to read information for more than one company in any given COI class.



Chinese Wall contd.

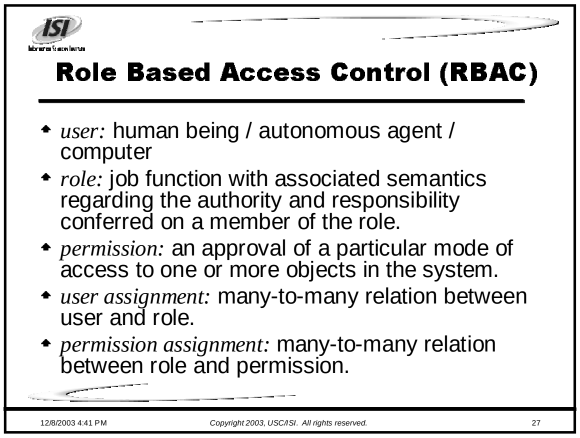
- An access request q=(object o, access right r, subject s) is granted iff all of the following properties are satisfied:
- Discretionary security property:
  - → The cell in the access matrix for row **s** and column **o** contains the requested right r.
- Mandatory security property:
  - → Subject **S** can access object **O** only if **O** is in the same company dataset as any object already read by **S**.

    or
  - → the object **O** does not belong to any of the COI classes of objects already accessed by subject **S**.



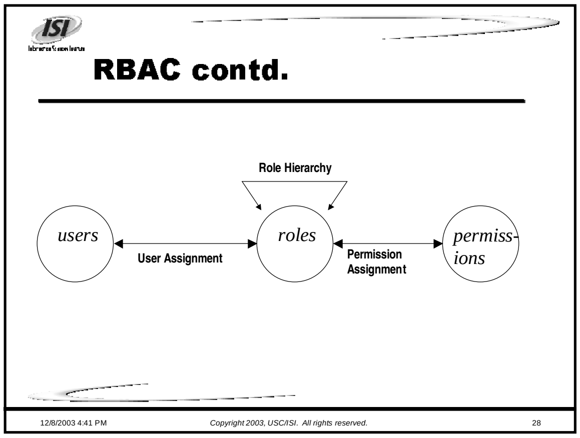
The Principal of Least Privilege

- "Each principal is given minimum access needed to accomplish its task"



Role Based Access Control (RBAC)

- *user:* human being / autonomous agent / computer
- *role:* job function with associated semantics regarding the authority and responsibility conferred on a member of the role.
- *permission:* an approval of a particular mode of access to one or more objects in the system.
- *user assignment:* many-to-many relation between user and role.
- *permission assignment:* many-to-many relation between role and permission.



RBAC contd.

Role Hierarchy

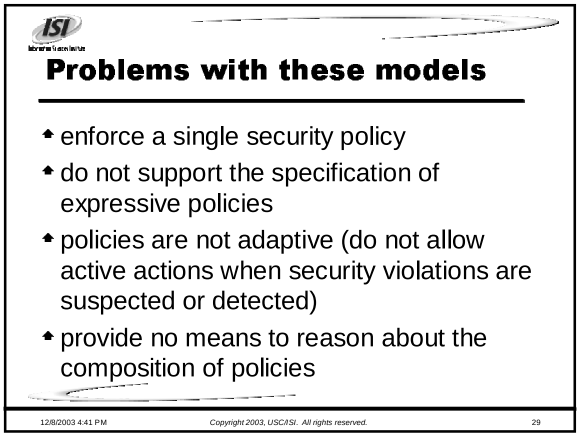
users

User Assignment

roles

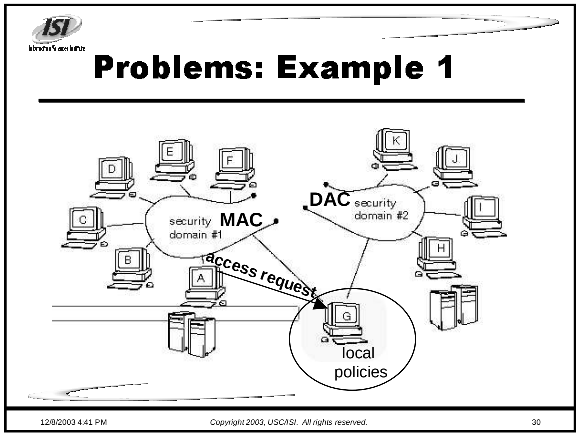
Permission Assignment

permiss-ions

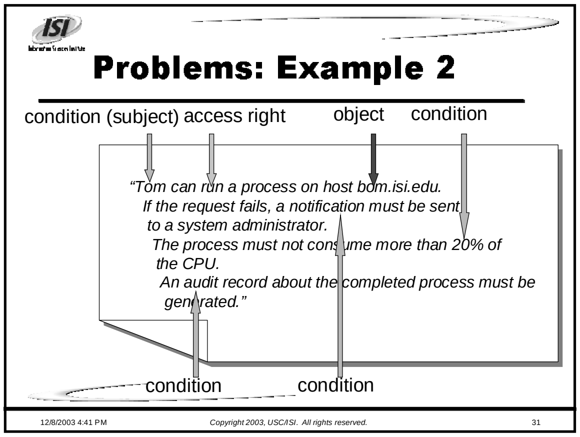


Problems with these models

- enforce a single security policy
- do not support the specification of expressive policies
- policies are not adaptive (do not allow active actions when security violations are suspected or detected)
- provide no means to reason about the composition of policies



Problems: Example 1

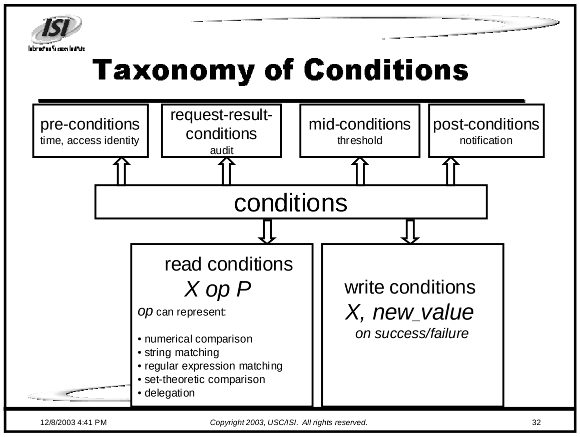


Problems: Example 2

condition (subject) access right object condition

"Tom can run a process on host bom.isi.edu. If the request fails, a notification must be sent to a system administrator. The process must not consume more than 20% of the CPU. An audit record about the completed process must be generated."

condition condition



Taxonomy of Conditions

pre-conditions
time, access identity

request-result-conditions
audit

mid-conditions
threshold

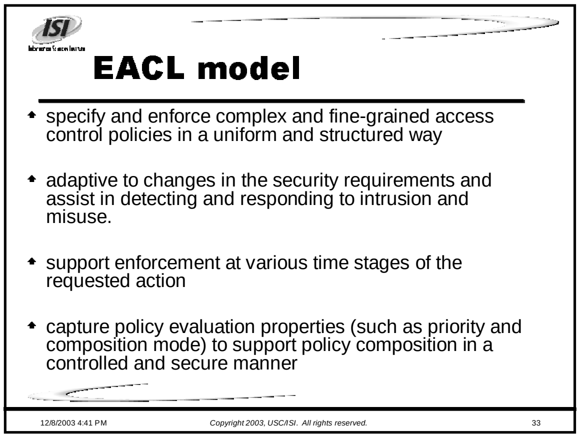
post-conditions
notification

conditions

read conditions
X op P
op can represent:

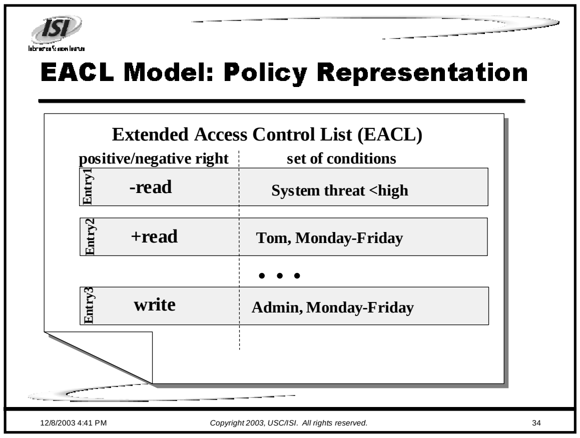
- numerical comparison
- string matching
- regular expression matching
- set-theoretic comparison
- delegation

write conditions
X, new_value
on success/failure



EACL model

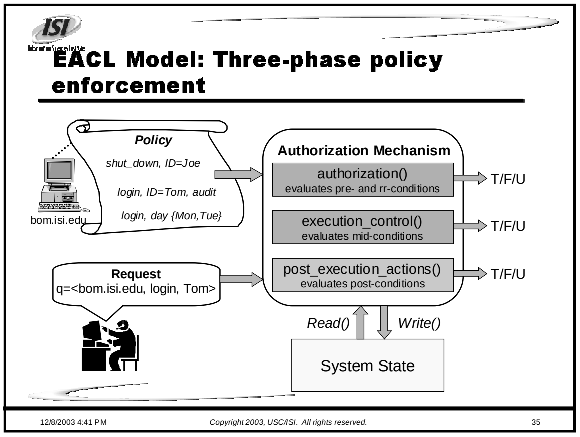
- specify and enforce complex and fine-grained access control policies in a uniform and structured way
- adaptive to changes in the security requirements and assist in detecting and responding to intrusion and misuse.
- support enforcement at various time stages of the requested action
- capture policy evaluation properties (such as priority and composition mode) to support policy composition in a controlled and secure manner



EACL Model: Policy Representation

Extended Access Control List (EACL)

	positive/negative right	set of conditions
Entry1	-read	System threat <high
Entry2	+read	Tom, Monday-Friday
	• • •	
Entry3	write	Admin, Monday-Friday



EACL Model: Three-phase policy enforcement

Policy
shut_down, ID=Joe
login, ID=Tom, audit
login, day {Mon,Tue}

bom.isi.edu

Request
q=<bom.isi.edu, login, Tom>

Authorization Mechanism

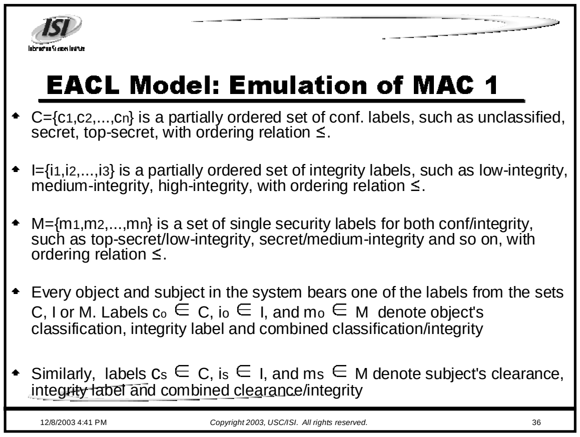
authorization()
evaluates pre- and rr-conditions → T/F/U

execution_control()
evaluates mid-conditions → T/F/U

post_execution_actions()
evaluates post-conditions → T/F/U

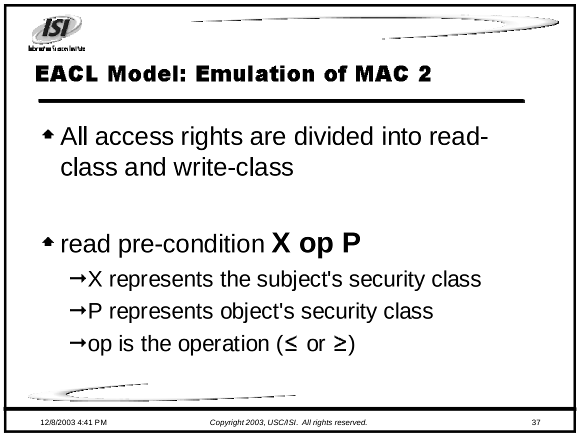
Read() *Write()*

System State



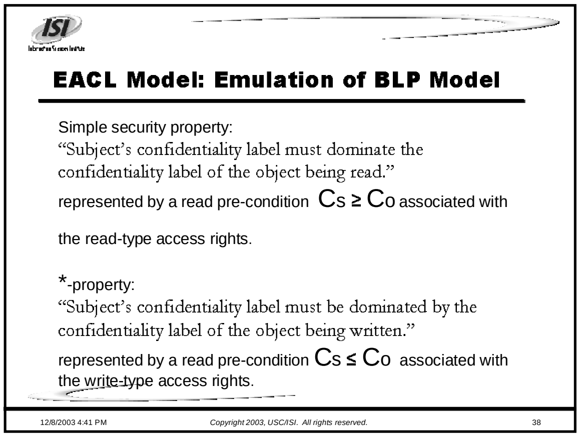
EACL Model: Emulation of MAC 1

- C={c_1,c_2,...,c_n} is a partially ordered set of conf. labels, such as unclassified, secret, top-secret, with ordering relation ≤.
- I={i_1,i_2,...,i_3} is a partially ordered set of integrity labels, such as low-integrity, medium-integrity, high-integrity, with ordering relation ≤.
- M={m_1,m_2,...,m_n} is a set of single security labels for both conf/integrity, such as top-secret/low-integrity, secret/medium-integrity and so on, with ordering relation ≤.
- Every object and subject in the system bears one of the labels from the sets C, I or M. Labels $c_o \in$ C, $i_o \in$ I, and $m_o \in$ M denote object's classification, integrity label and combined classification/integrity
- Similarly, labels $c_s \in$ C, $i_s \in$ I, and $m_s \in$ M denote subject's clearance, integrity label and combined clearance/integrity



EACL Model: Emulation of MAC 2

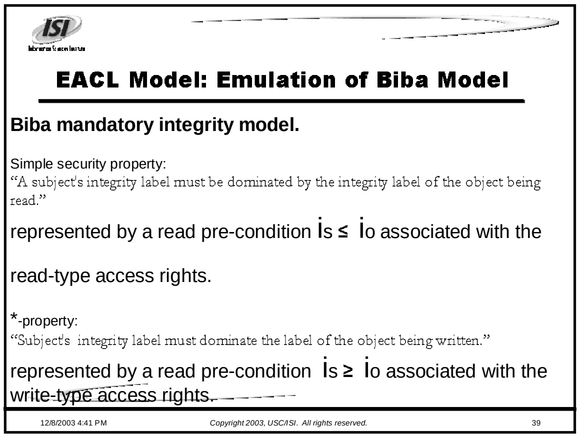
- All access rights are divided into read-class and write-class
- read pre-condition **X op P**
  - →X represents the subject's security class
  - →P represents object's security class
  - →op is the operation (≤ or ≥)



EACL Model: Emulation of BLP Model

Simple security property:
"Subject's confidentiality label must dominate the confidentiality label of the object being read."
represented by a read pre-condition $C_S \ge C_O$ associated with the read-type access rights.

*-property:
"Subject's confidentiality label must be dominated by the confidentiality label of the object being written."
represented by a read pre-condition $C_S \le C_O$ associated with the write-type access rights.

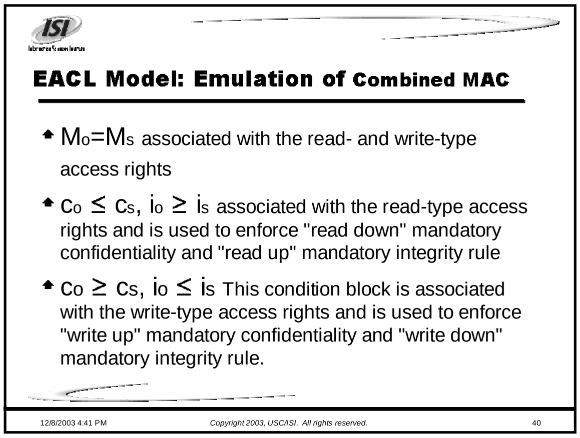


EACL Model: Emulation of Biba Model

Biba mandatory integrity model.

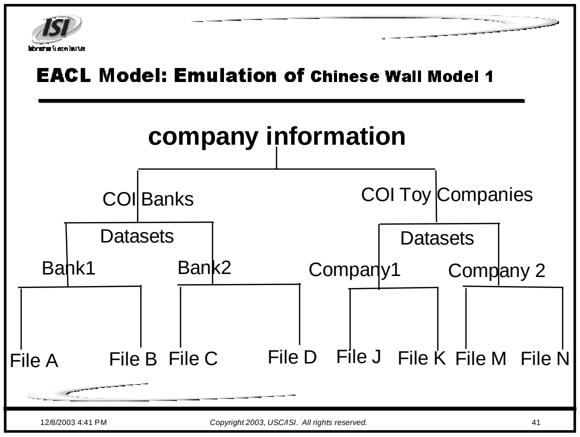
Simple security property:
"A subject's integrity label must be dominated by the integrity label of the object being read."
represented by a read pre-condition $i_S \le i_O$ associated with the read-type access rights.

*-property:
"Subject's integrity label must dominate the label of the object being written."
represented by a read pre-condition $i_S \ge i_O$ associated with the write-type access rights.



EACL Model: Emulation of Combined MAC

- $M_O = M_S$ associated with the read- and write-type access rights
- $c_O \le c_S$, $i_O \ge i_S$ associated with the read-type access rights and is used to enforce "read down" mandatory confidentiality and "read up" mandatory integrity rule
- $c_O \ge c_S$, $i_O \le i_S$ This condition block is associated with the write-type access rights and is used to enforce "write up" mandatory confidentiality and "write down" mandatory integrity rule.

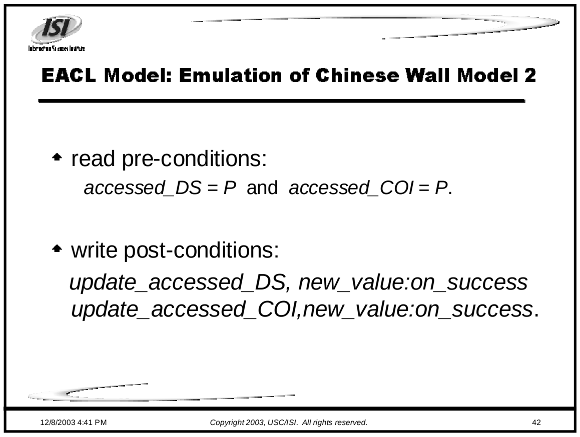


EACL Model: Emulation of Chinese Wall Model 1

company information

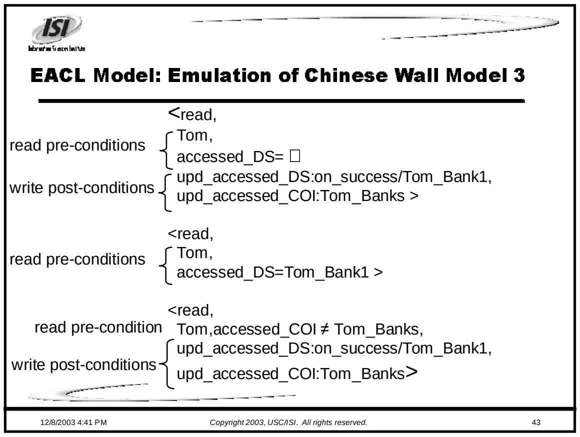
COI Banks
Datasets
Bank1
File A
File B
Bank2
File C
File D

COI Toy Companies
Datasets
Company1
File J
File K
Company 2
File M
File N



EACL Model: Emulation of Chinese Wall Model 2

- read pre-conditions:
  accessed_DS = P and *accessed_COI = P*.
- write post-conditions:
  update_accessed_DS, new_value:on_success
  update_accessed_COI,new_value:on_success.



EACL Model: Emulation of Chinese Wall Model 3

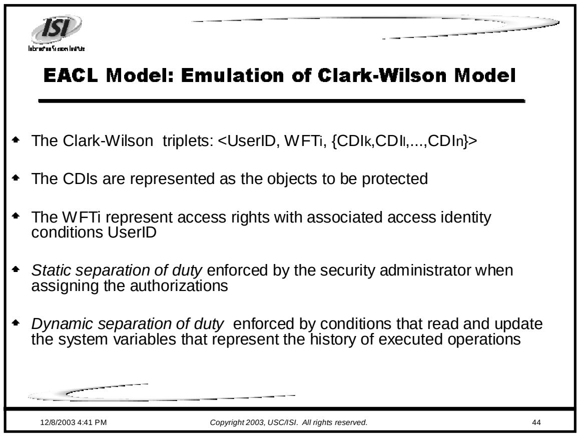
<read,
read pre-conditions { Tom,
accessed_DS= □
write post-conditions { upd_accessed_DS:on_success/Tom_Bank1,
upd_accessed_COI:Tom_Banks >

<read,
read pre-conditions { Tom,
accessed_DS=Tom_Bank1 >

<read,
read pre-condition Tom,accessed_COI ≠ Tom_Banks,
write post-conditions { upd_accessed_DS:on_success/Tom_Bank1,
upd_accessed_COI:Tom_Banks>

12/8/2003 4:41 PM

43

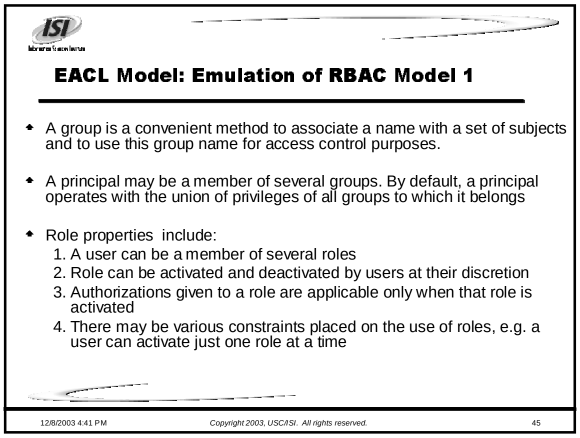


EACL Model: Emulation of Clark-Wilson Model

- The Clark-Wilson triplets: <UserID, WFTi, {CDIk,CDIl,...,CDIn}>
- The CDIs are represented as the objects to be protected
- The WFTi represent access rights with associated access identity conditions UserID
- *Static separation of duty* enforced by the security administrator when assigning the authorizations
- *Dynamic separation of duty* enforced by conditions that read and update the system variables that represent the history of executed operations

12/8/2003 4:41 PM

44

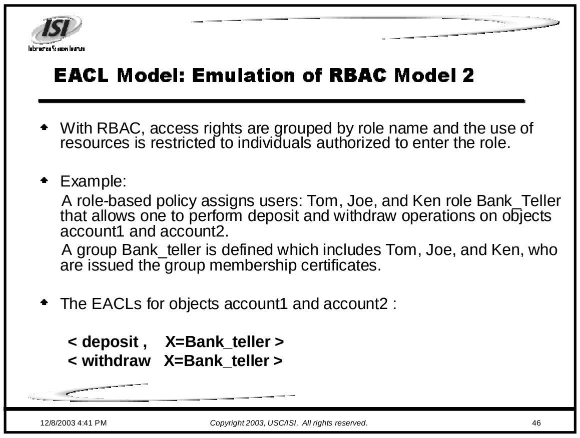


EACL Model: Emulation of RBAC Model 1

- A group is a convenient method to associate a name with a set of subjects and to use this group name for access control purposes.
- A principal may be a member of several groups. By default, a principal operates with the union of privileges of all groups to which it belongs
- Role properties include:
  1. A user can be a member of several roles
  2. Role can be activated and deactivated by users at their discretion
  3. Authorizations given to a role are applicable only when that role is activated
  4. There may be various constraints placed on the use of roles, e.g. a user can activate just one role at a time

12/8/2003 4:41 PM

45



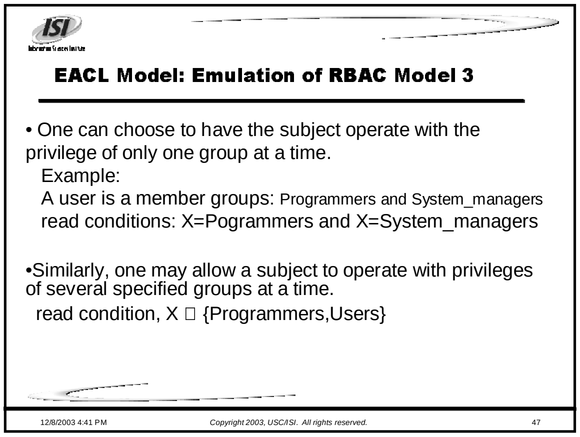
EACL Model: Emulation of RBAC Model 2

- With RBAC, access rights are grouped by role name and the use of resources is restricted to individuals authorized to enter the role.
- Example:
  A role-based policy assigns users: Tom, Joe, and Ken role Bank_Teller that allows one to perform deposit and withdraw operations on objects account1 and account2.
  A group Bank_teller is defined which includes Tom, Joe, and Ken, who are issued the group membership certificates.
- The EACLs for objects account1 and account2 :

< deposit , X=Bank_teller >
< withdraw X=Bank_teller >

12/8/2003 4:41 PM

46



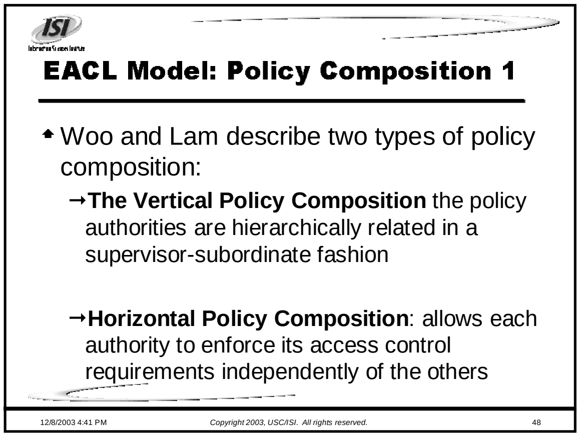
EACL Model: Emulation of RBAC Model 3

• One can choose to have the subject operate with the privilege of only one group at a time.
Example:
A user is a member groups: Programmers and System_managers
read conditions: X=Pogrammers and X=System_managers

•Similarly, one may allow a subject to operate with privileges of several specified groups at a time.
read condition, X □ {Programmers,Users}

12/8/2003 4:41 PM

47

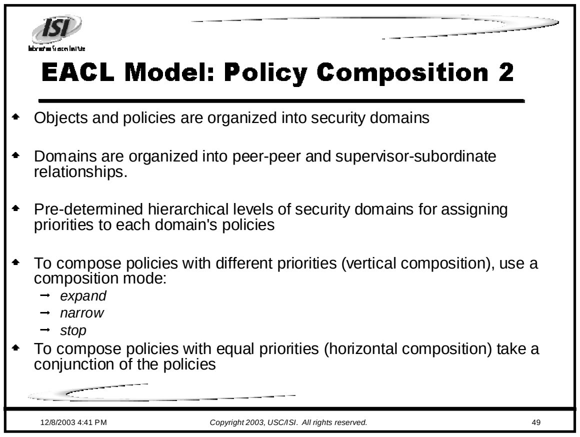


EACL Model: Policy Composition 1

- Woo and Lam describe two types of policy composition:
  - →**The Vertical Policy Composition** the policy authorities are hierarchically related in a supervisor-subordinate fashion
  - →**Horizontal Policy Composition**: allows each authority to enforce its access control requirements independently of the others

12/8/2003 4:41 PM

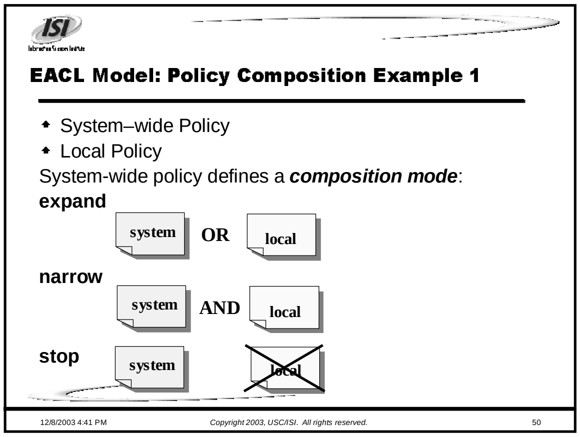
48



EACL Model: Policy Composition 2

- Objects and policies are organized into security domains
- Domains are organized into peer-peer and supervisor-subordinate relationships.
- Pre-determined hierarchical levels of security domains for assigning priorities to each domain's policies
- To compose policies with different priorities (vertical composition), use a composition mode:
  - → *expand*
  - → *narrow*
  - → *stop*
- To compose policies with equal priorities (horizontal composition) take a conjunction of the policies

12/8/2003 4:41 PM Copyright 2003, USC/ISI. All rights reserved. 49



EACL Model: Policy Composition Example 1

- System–wide Policy
- Local Policy

System-wide policy defines a ***composition mode***:

expand

system **OR** local

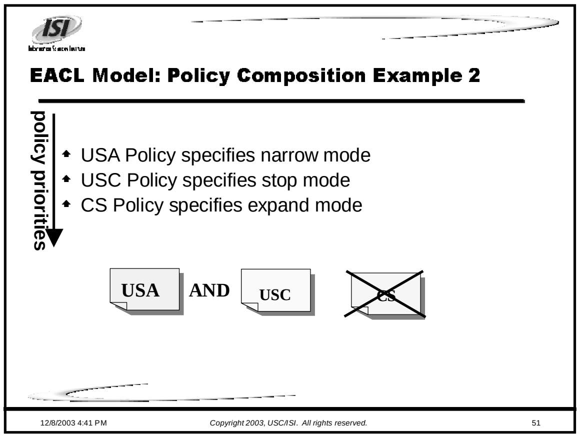
narrow

system **AND** local

stop

system ~~local~~

12/8/2003 4:41 PM Copyright 2003, USC/ISI. All rights reserved. 50



EACL Model: Policy Composition Example 2

policy priorities

- USA Policy specifies narrow mode
- USC Policy specifies stop mode
- CS Policy specifies expand mode

USA **AND** USC ~~CS~~

12/8/2003 4:41 PM Copyright 2003, USC/ISI. All rights reserved. 51