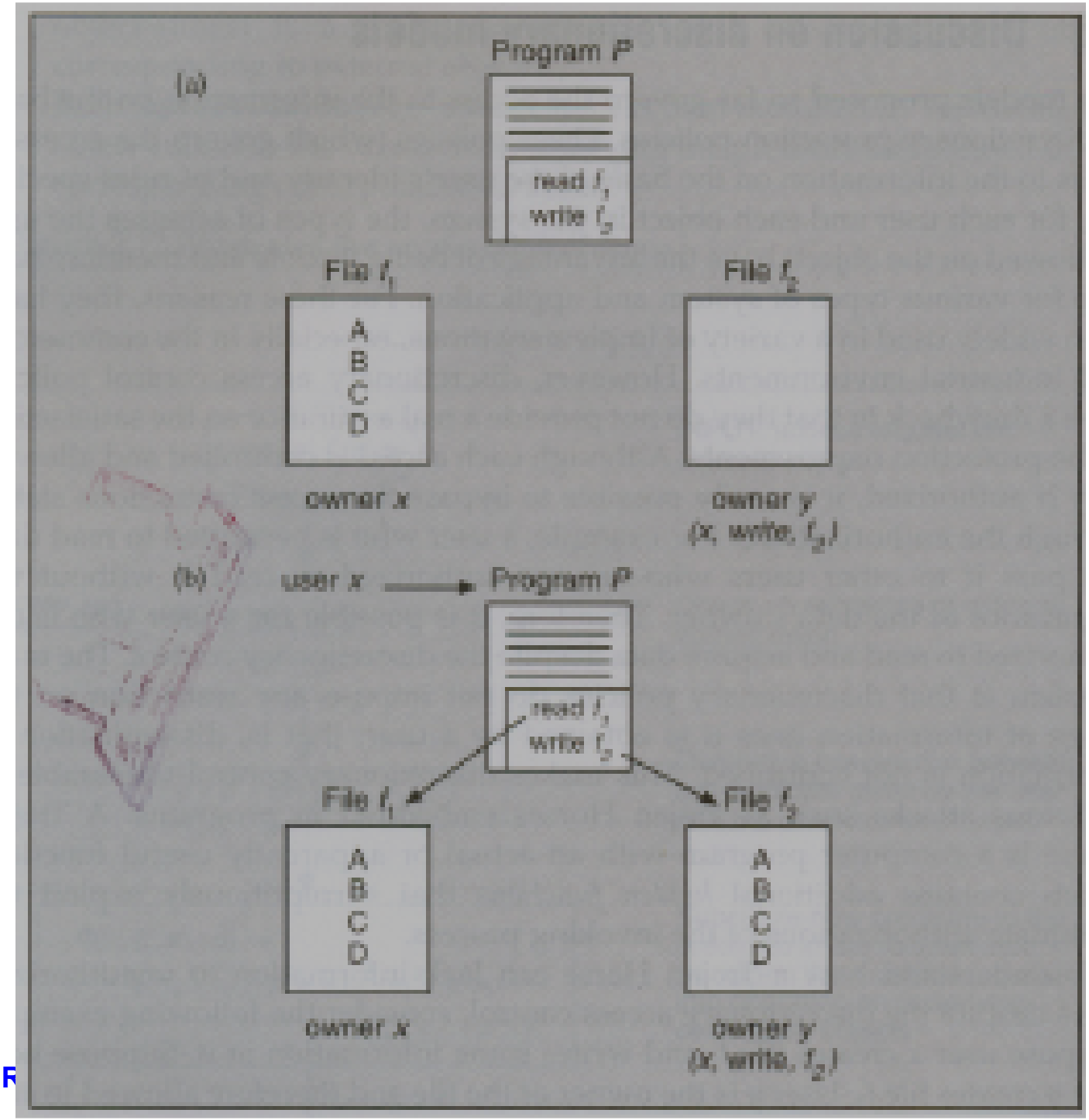
Discretionary Vs. Mandatory

- Discretionary access controls (DAC)
  - Privilege propagated from one subject to another
  - Possession of an access right is sufficient to access the object
- Mandatory access controls (MAC)
  - Restrict access on the basis of security labels

Inherent Weakness of DAC

- Unrestricted information flow from one object to another
  - Ali owns file A and grants read to Taghi
  - Taghi reads file A and write to file B
  - Ali has no control over file B
- Suppose users can be trusted
  - How about software? Trojan horses


(a)
Program P
read f_1
write f_2
File f_1
A
B
C
D
owner x
File f_2
owner y
(x, write, f_2)
(b)
user x
Program P
read f_1
write f_2
File f_1
A
B
C
D
owner x
File f_2
A
B
C
D
owner y
(x, write, f_2)

Military Security Requirements

- Protect data from unauthorized modification
- Protect data from unauthorized access
- Strong information flow control
- Protect system from attacks resulting in a loss of service

Mandatory Access Controls

- Enforce a label-based policy:
  - Assign security levels to all data
  - Assign a security clearance to each user
  - DBMS should make sure that all users have access to only those data for which they have a clearance

Mechanisms —Reference Monitor

- It contains security classes of all objects and subjects.
- Whenever a subject accesses an object, it must do so via the reference monitor.
- It enforces the two model restrictions faithfully.
- It is always running, cannot be bypassed, and cannot be tampered with.

MANDATORY ACCESS CONTROL MODELS

Bell-LaPadula Model

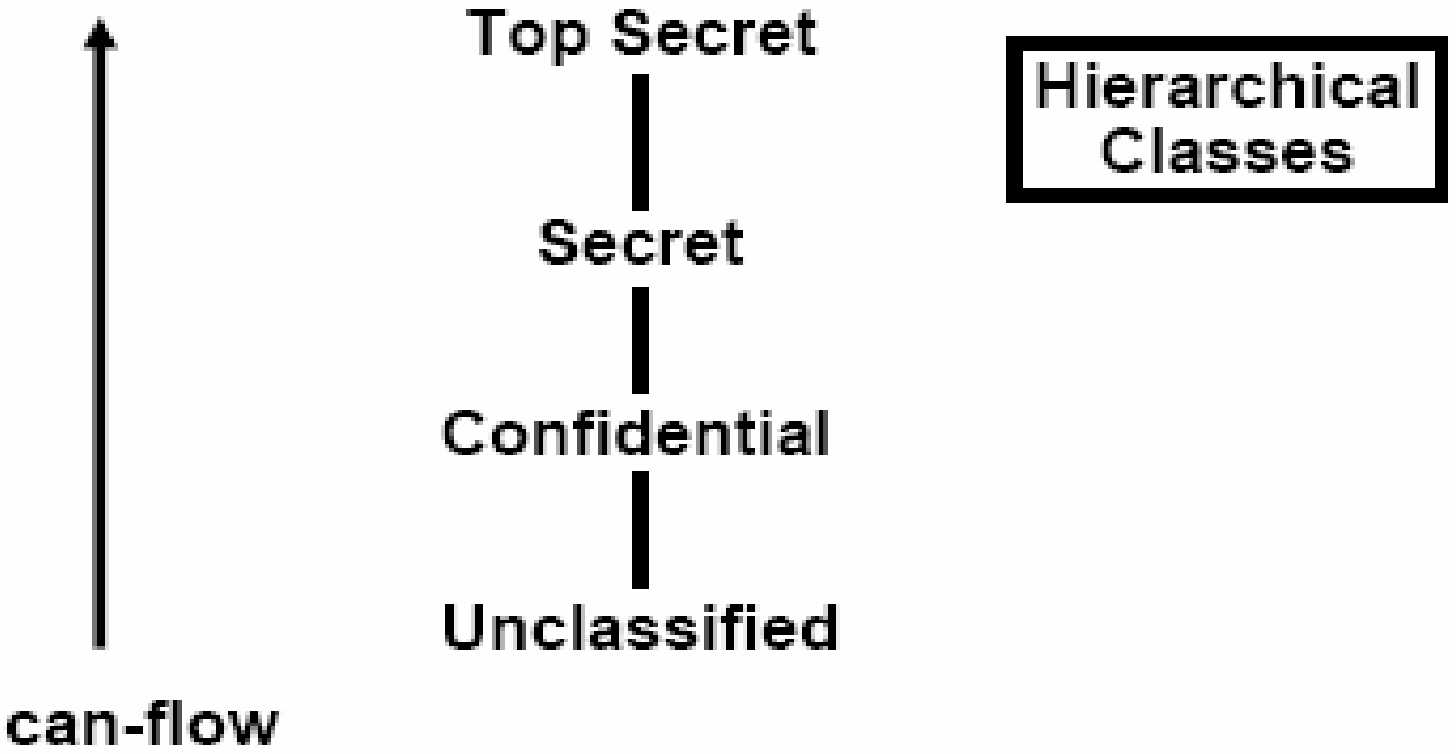
- Bell and Lapadula , 1976 is an extension of the access matrix model oriented to the definition of the security requirements in complex systems where system elements can be classified.
- Initially oriented to protection in the OS env, faces the problem of information secrecy.
- Each security level is described by two components: a classification and a set of categories.
- The classification is an element of the set TS, S, C, and U. TS>S>C>U.
- The set of categories is a subset of non-hierarchical set of elements; may be an application or organization descriptor (e.g., financial, educational, etc.)

Bell-LaPaduala Model (cont.)

- **Classification Levels for subjects/ programs and objects/resources:**
  - **1-Top secret**
  - **2- Secret**
  - **3- Confidential**
  - **4- Unclassified**
- **Security Levels: $L_1 = (C_1, S_1)$, $L_2 = (C_2, S_2)$**
- **S_1 and S_2 are categories.**
- **L_1 is higher or equal to L_2, if and only if:**
- **$C_1 > C_2$**
- **$S_1 \supseteq S_2$**

- Lattice of the Security levels in the next slides

Lattice Structures



- reflexive and transitive edges are implied but not shown

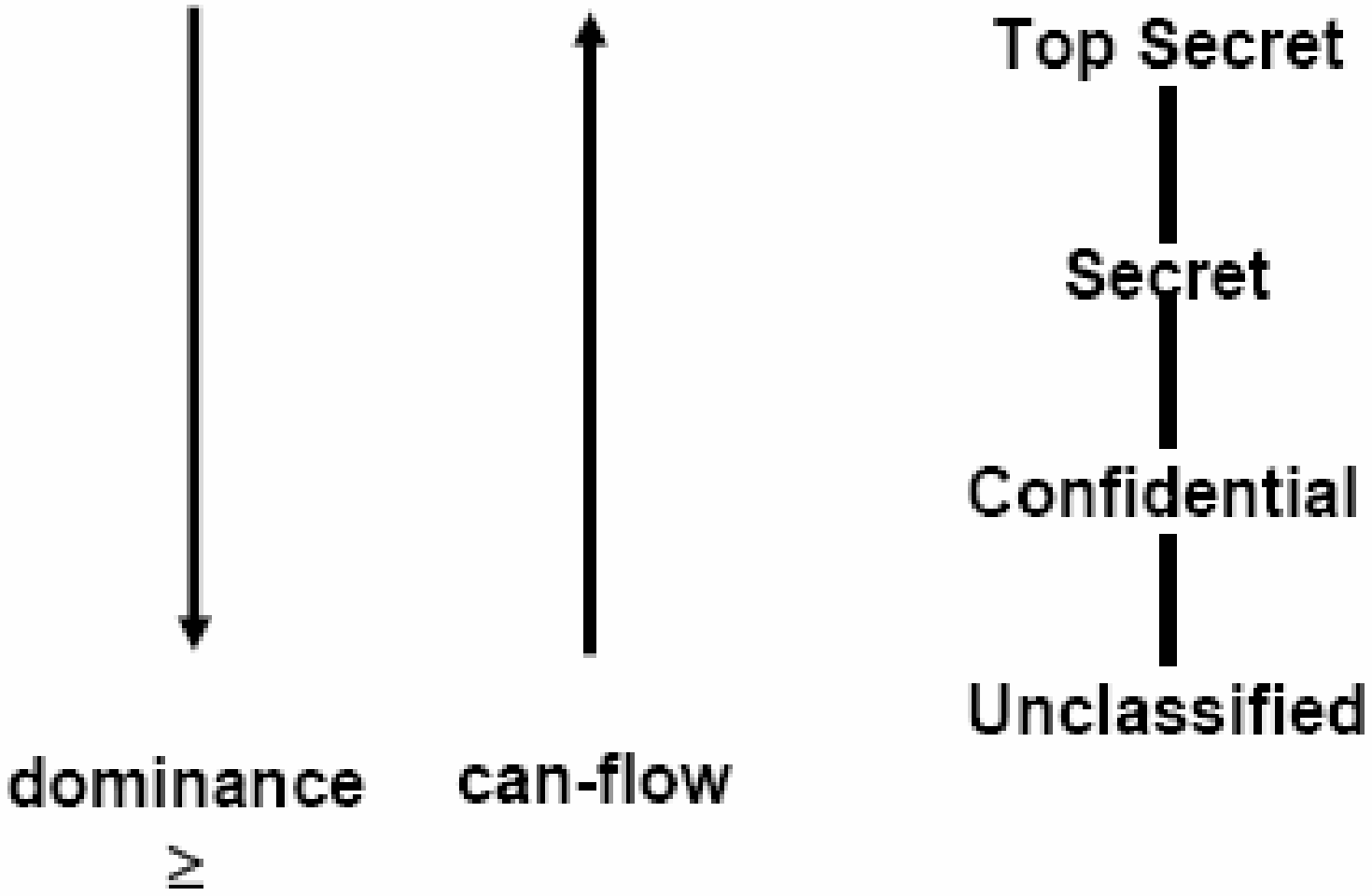
Partial Order

- A relation *R* Defined over S_C X S_C
  - Reflexive: for all A in S_C, (A, A) in *R*
  - Anti-symmetric: if (A,B) and (B,A) are in *R*, then A=B
  - Transitive: (A,B) and (B,C) are in *R*, then (A,C) in *R*
- If (A,B) in *R*, we say B dominate A
  - $B \geq A$

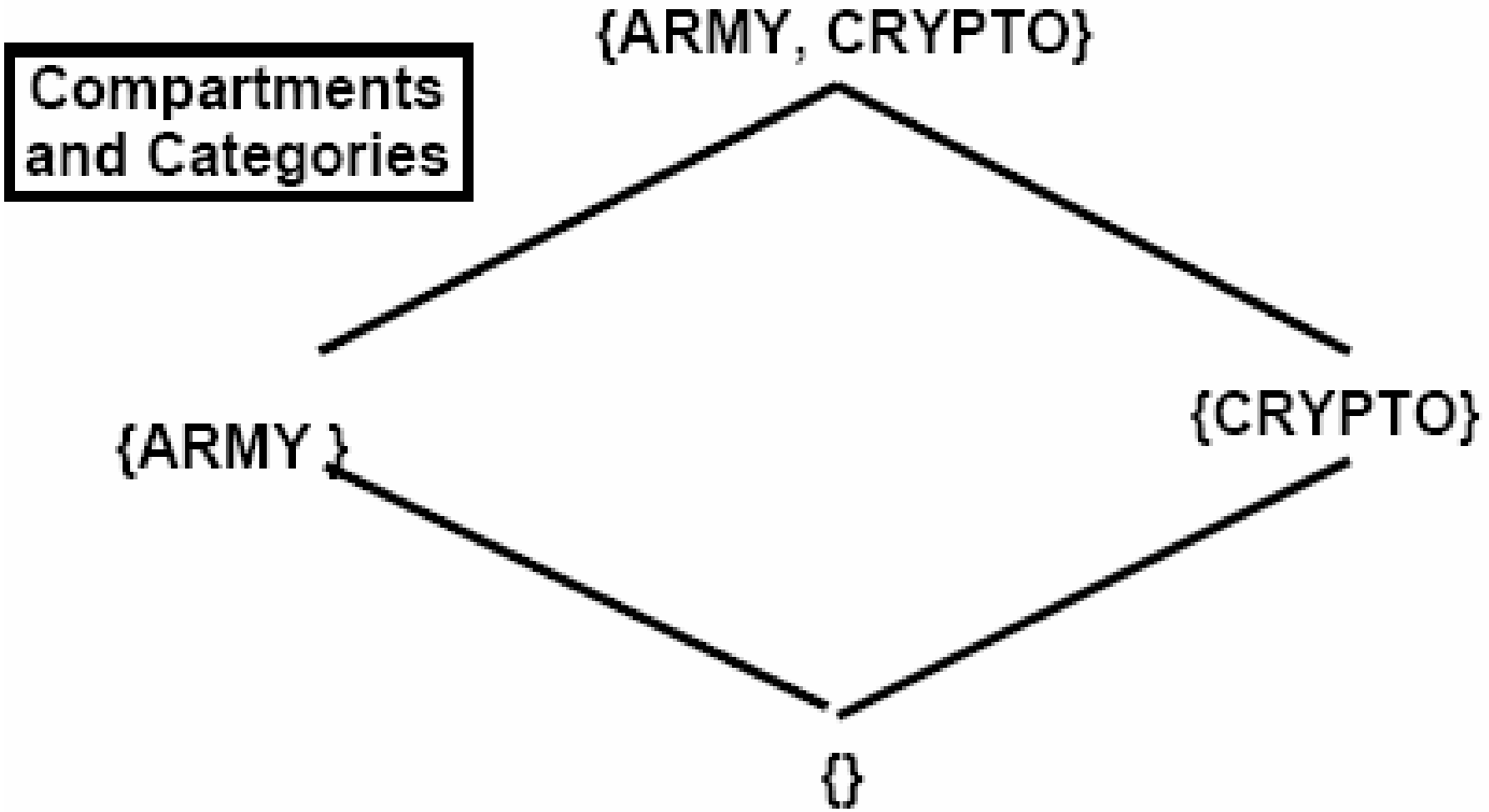
Partial Order

- Partial order among security labels
  - $B \geq A$, then information can flow from A to B
  - If $B \geq A$ but not $A \geq B$ (i.e., A !=B), then information cannot flow from B to A

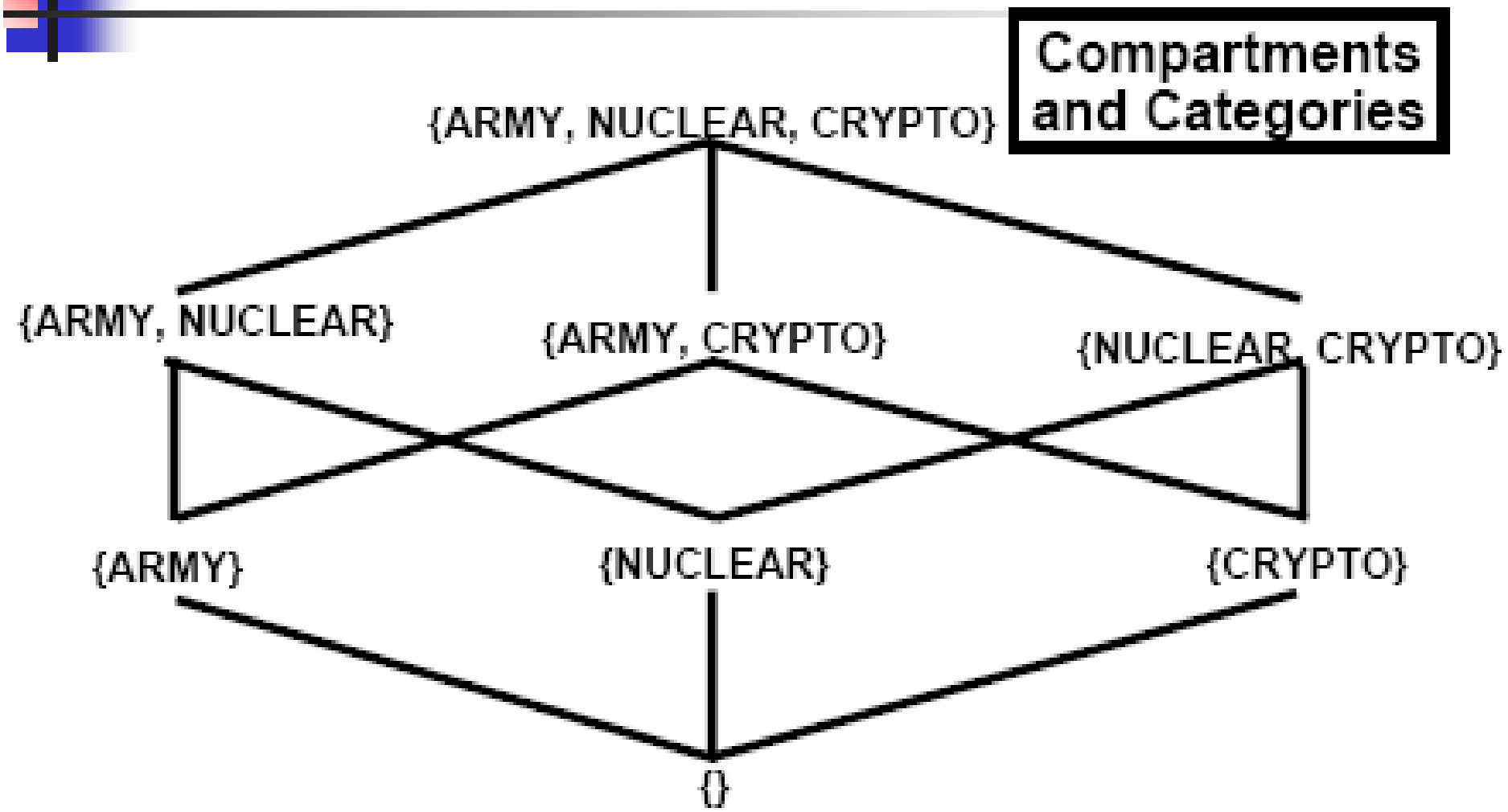
Lattice Structures



Lattice Structures

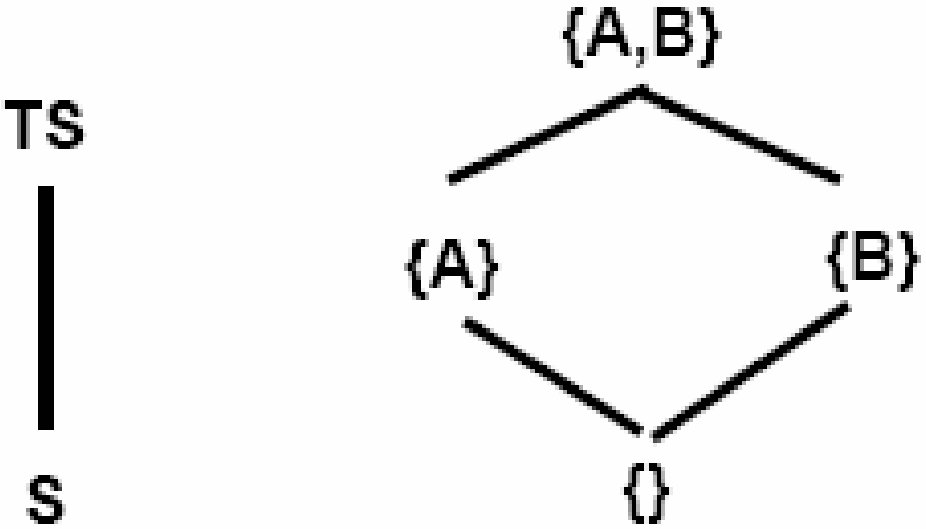


Lattice Structures



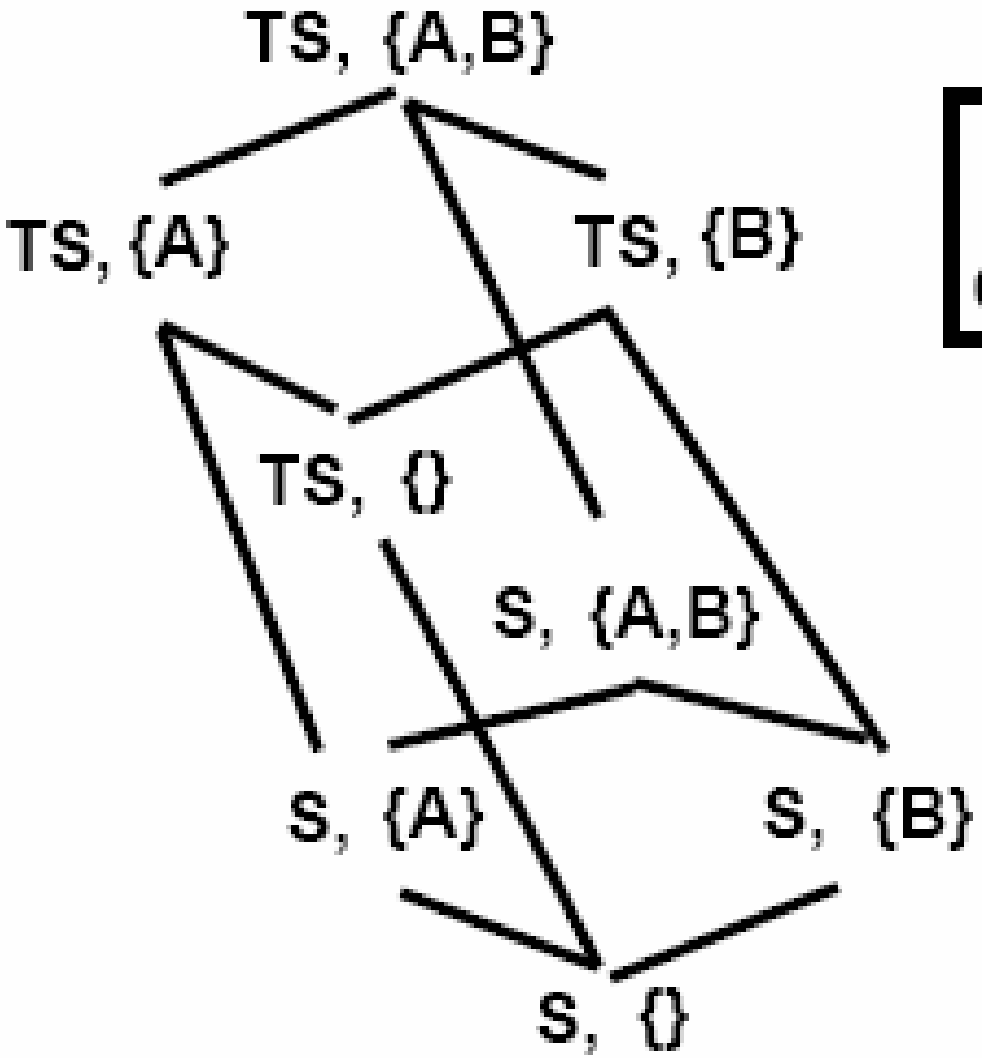
Lattice Structures

Hierarchical Classes with Compartments



product of 2 lattices is a lattice

Lattice Structures



Hierarchical Classes with Compartments

Bell-LaPaduala Model (cont.)

- The model is based on the subject-object paradigm: subjects are active elements that can execute actions; objects are passive elements that can contain information.
- Subjects are processes acting on behalf of users, who are assigned a sec level called clearance. This is the level of trust to the user. It is assumed that the user does not disclose info to those not holding the appropriate clearance.
- The sensitivity of the info stored in objects, specifies their sec level.

- Model considers 4 access modes executable by subjects on objects:
  - **Read-only or Read**
  - **Append (writing without reading)**
  - **Execute (executes an object /program)**
  - **Read-write or Write**
- The creator of an object is considered as its owner and so is allowed to grant and revoke authorization on the object to other users.
- The ownership is not transferable.

The System State

- **The state of the system is described by the 4-tuple (b,M, f, H), where:**
- **b**: the current access set; (*s*, *o*, *m*): subject *s* has current access mode *m* on object *o*.
- **M**: the access matrix; M (*s*, *o*)- same as the access/matrix model- specified the access modes subject s is authorized for an object o.
- **f**: a level function that associates with each subject & object in the system its security level

$$f: O \cup S \rightarrow L$$

  - To each object only one sec level is associated.

- To each user, 2 sec levels are associated: its clearance (f_s) (the sec level assigned when it is created) and its current level (f_c) (the sec level at which the user is actually operating. It may change during the subject's life time, but always: $f_c(s) \leq f_s(S)$
- This means that the user can log into the system at any sec level which is dominated by the user clearance.

- **H**: The current objects hierarchy. This is needed to specify the dominance relationship between security levels.

Bell-Lapaduala Model (cont.)

The system state can be changed by executing operations. The operations are:

- **Get access** : initiate access to object in a given mode. The execution of this operation modifies the triple <s,o,m>
- **Release access** : terminate access previously started by get. The triple is removed as the current access.
- **Give access**: grant an access mode on an object to a subject. This operation changes the access matrix by inserting the access mode being granted. If the insertion respects the axioms of the mandatory policy.
- **Rescind access** : revoke access previously granted with the “give” operation.

Bell-Lapaduala Model (cont.)

Operations - contd.:

- **Create object**: an object may be inactive or active; this operation takes an inactive object and adds to the object hierarchy.
- **Delete object**: deactivates an active object from the hierarchy.
- **Change subject security level,** changes f.
- **Change object security level,** changes f. Only on the inactive objects, and only upgrading the object security level. The new security level of the object must be dominated by the subject requesting the change clearance.

Bell-Lapaduala Model: Axioms

A system state is secure, iff it satisfies a set of properties. This is controlled by a reference monitor.

A trusted subject is a subject that can be relied on not to compromise security.

Some constraints are enforced only on requests made by untrusted subjects.

(1) Simple security (ss) property:

- A subject may have read or write access to an object only if the clearance of the subject dominates the security level of the object.
- *A system state v= (b, M, f, H) satisfies the ss-property iff for each M[s,o] containing the read or write access modes, $f_s(s) \geq f_o(o)$.*
- *The aim of this property is to prevent subjects from **reading** info having the higher classification than the subject's clearance.*

BLP Axioms: Star (*) property

- An untrusted subject may have append access to an object, if the security level of the object dominates the security level of the subject. ($f_o(o) \geq f_c(s)$)
- An untrusted subject may have write access to an object if only the security level of the object is equal to the current security level of the subject. ($f_o(o) = f_c(s)$)
- An untrusted subject may have read access to an object only if the security level of the object is dominated by the current security level of the subject. $f_c(s) \geq f_o(o)$.

BLP Axioms: Star (*) property

- A system state v= (b,M,f,H) satisfies the *-property iff $\forall s \in S'$ (where $S' \subseteq S$) is the set of untrusted subjects and $\forall o \in O$:
  - append $\in M[s,o] \Rightarrow f_c(s) \leq f_o(o)$.
  - *write* $\in M[s,o] \Rightarrow f_c(s) = f_o(o)$.
  - *read* $\in M[s,o] \Rightarrow f_c(s) \geq f_o(o)$.
- *Satisfaction of the *-property includes the ss-property. Note that $f_s(s) \geq f_c(s)$.*
- *Both the ss-property and the *-property are required. ss-property is needed for all subjects; while the *-property is required for untrusted subjects.*

BLP summary

- *SUMMARY IN SIMPLER TERMS; two basic principles*:
  - a subject can only read objects whose sec level is dominated by the level of t he subject $f_s(s) \geq f_o(o)$ .. NO READ-UP SECRECY.
  - a subject can only write objects whose sec level dominates the level of t he subject $f_s(s) \leq f_o(o)$ .. NO WRITE-DOWN SECRECY.
- These principles have been adopted by all MAC policies, as they controls the flow of info ensuring info not to be accessible by subjects not having the necessary clearance.

BLP summary (cont.)

(3) Tranquility principle

No subject can modify the classification of an active object

(4) Discretionary property (ds-property)

Every current access must be present in the access matrix: that is, a subject can exercise only accesses for which it has the **necessary authorization**. A system state satisfies the discretionary property if and only if for all subjects s, objects o, and access mode m:

$$< s, o, m> \in b \Rightarrow \; m \in M[s,o]$$

BLP extension

(5) Non-accessibility of inactive objects

A subject cannot read the contents of inactive object

(6) Rewriting of inactive objects

A newly activated object is assigned an initial state independent of the previous activations of the object.

Although proposed back in 1976, this model still dominates the design of the secure trusted database implementations by vendors for military/government applications following Mandatory Access Control.

Biba model

- BLP and other information-flow based security definitions address confidentiality, what about integrity
- BLP does not protect the system from unauthorized modification of info.
- Biba (1977) model is aimed to protect info integrity
- What does integrity mean?
  - data not changed in “incorrect” ways
- Difference between confidentiality & integrity
  - a subject cannot leak a piece of confidential information without reading it, but can introduce low-integrity information without reading any
  - some trust has to be placed on subjects for integrity.

Biba Model …

- The bases of the Biba model is similar to that of BLP.
- Notation of Subject and Objects and their classification.
- Subjects are active entities
- Objects are passive entities
- Each O and S is assigned a classification named an integrity level.
- The integrity level assigned to an object reflects both the degree of trust that can be placed on the object info and the potential damage that could result from unauthorized modification of the info.

Biba

- Each integrity level has two elements:
  - A classification
    - Crucial
    - Very Important
    - Important
    - C > VI > I
  - A set of categories
    - A subset of a non-hierarchical set
- Integrity levels form a lattice with respect to the Partial Order dominance relationship ($\geq$).

Biba

- L1 = (C1, S1) , L2= (C2, S2)
- L1 $\geq$ L2 *iff* C1 $\geq$ C2 && S1 $\supseteq$ S2

- L1 > L2 iff C1 > C2 && S1 $\supset$ S2

- Incompatibility:
  - neither L1 $\geq$ L2 nor L2 $\geq$ L1
- The integrity level of a subject is its belonging user's integrity level reflect the user trustworthiness for inserting, deleting, and modifying info.
- The integrity level of an object indicates the level of trust that can be placed in the object and the potential damage that could result from unauthorized modification of the info.

Biba: Access modes

The model considers the access modes:

- Modify: to write info in an object, similar to write.
- Invoke: Allows subjects to communicate
- Observe: to read info from an object.
- Execute: to execute an object (program)

- No access mode for administration of authorization: Ownership is not considered in the Biba model.

Axioms

- No administration operation for grant or revoke in the model. Direct modification of the ACLs associated with the objects.

- A family of policies are proposed in Biba, each adopting different conditions to ensure info integrity. Two major groups: non-discretionary and discretionary policies.

Non-Discretionary Policies

- Determines accesses executable by the subjects on the objects basis on their security levels; includes the following policies:

1. Low-watermark policy for subjects; axioms:
   - A subject can hold the **modify** access to an object if its integrity level ≥ that of the object.
   - A subject can hold the **invoke** access to another subject if the integrity level of the first subject ≥ that of the other subject.

– A subject can hold the **observe** access to whatever object. After observing of a subject on an object, its integrity level = the greatest lower bound between the subject and the object integrity level (before the operation).

– This policy is said to be dynamic; since it possibly decreases the integrity level of a subject upon each observing objects with lower or incompatible integrity levels.

– Main drawback: order of the operation yields different results! Accessing after observe or before observing an object differs!

2- Low-watermark policy for objects

- A subject can hold the modify access to objects at whatever integrity level. After each modify access of S on O, the integrity level of O is set to the greatest lower bound between the integrity levels of S and O (before access).
- The policy is dynamic. The integrity level of objects is decreased based on the modifier.
- Allows improper modifications! When a modification is done, the int level of the object is downgraded, but exposing to threats. The scenario may not be recoverable!

3- Low-watermark integrity audit policy:

- A subject can modify objects at whatever integrity level. If S modifies O at higher or incompatible integrity level, the sec violation is recorded in an audit trail.
- The policy does not prevent improper modifications.

4- Ring policy: Int. levels of subjects and objects are fixed during their lifetime. It is based on the following axioms:

  - S can hold the **modify** access to O if the int. level of S $\geq$ the int. level of O.
  - S can hold the **invoke** access to S' if the int. level of S $\leq$ the int. level of S'!
  - S can hold the **observe** access to objects of whatever int. level.

- Improper modification can occur indirectly. A high-level subject can observe an object (at whatever level) and then modify an object at its own int level.

5- Strict integrity policy:

- Integrity *-property: S can hold the **modify** access to O if the int level of S $\geq$ the int level of O.
- Invocation policy: S can hold the **invoke** access to S' if the int level of S $\geq$ the int level of S'.
- Simple integrity condition: S can hold the **observe** access to O only if its int level $\leq$ the int level of O.

- The policy prevents from the transfer of info from low-integrity object to objects at higher int levels.

- Summary of the Strict integrity policy:
  - No Read-Down integrity
  - No Write-Up integrity

Discretionary Policies in Biba

- Access Control Lists: Each object has an access control list indicating the subjects that can access the object. Modification of the ACL can be done by subjects who have the modify access on that object.

- Objects hierarchy: Organization of objects through a rooted tree structure: To access an object, a subject must have **observe** access to all its ancestors.

- Ring: Each subject is assigned a privilege attribute (ring) Low number rings are higher privileged. Accesses are allowed based on the allowed range of rings.
  - A subject can hold the modify access on objects in an allowed range of rings.
  - A subject can hold the invoke access to subjects of greater privilege only in an allowed range of rings. A subject can hold the invoke access on any subject with lower or equal privilege.
  - A subject can hold the observe access on objects in an allowed range of rings.

Summary of Biba's Models

- Different models assume different kinds of trust in subjects
- the ring model assumes subjects can correctly process inputs and generate data of a certain integrity level
- the low-water mark models assume subjects do not introduce low integrity information themselves, but may be contaminated by the source
- the strict MAC model assumes subjects may be contaminated by the source and can only generate data of a certain integrity level

Key Difference between Confidentiality and Integrity

- Noninterference policies for protecting data confidentiality, constraining:
  - Who can read the secret data.
  - Where the secret data will flow to (in the future).
- Dually, integrity policies constrain:
  - Who can write to the data.
  - Where the data is derived from (in the past, the history of the data)."

Key Difference between Confidentiality and Integrity ?????

- For confidentiality, no trust needs to be placed on subjects
  - one does need trusted subjects to make system realistic, but they are not needed for confidentiality
- For integrity, one has to trust subjects
  - therefore; one has to justify such trust