Access Control: DAC and MAC/LBAC

Prof. Ravi Sandhu
Executive Director and Endowed Chair

Lecture 2

ravi.utsa@gmail.com
www.profsandhu.com

Authentication, Authorization, Audit

AAA

Authentication	Authorization	Audit
Who are You?	What are You Allowed to Do?	What Did You Do?

siloed → integrated

Access Control

Fixed policy

Discretionary Access Control (DAC), 1970

Mandatory Access Control (MAC), 1970

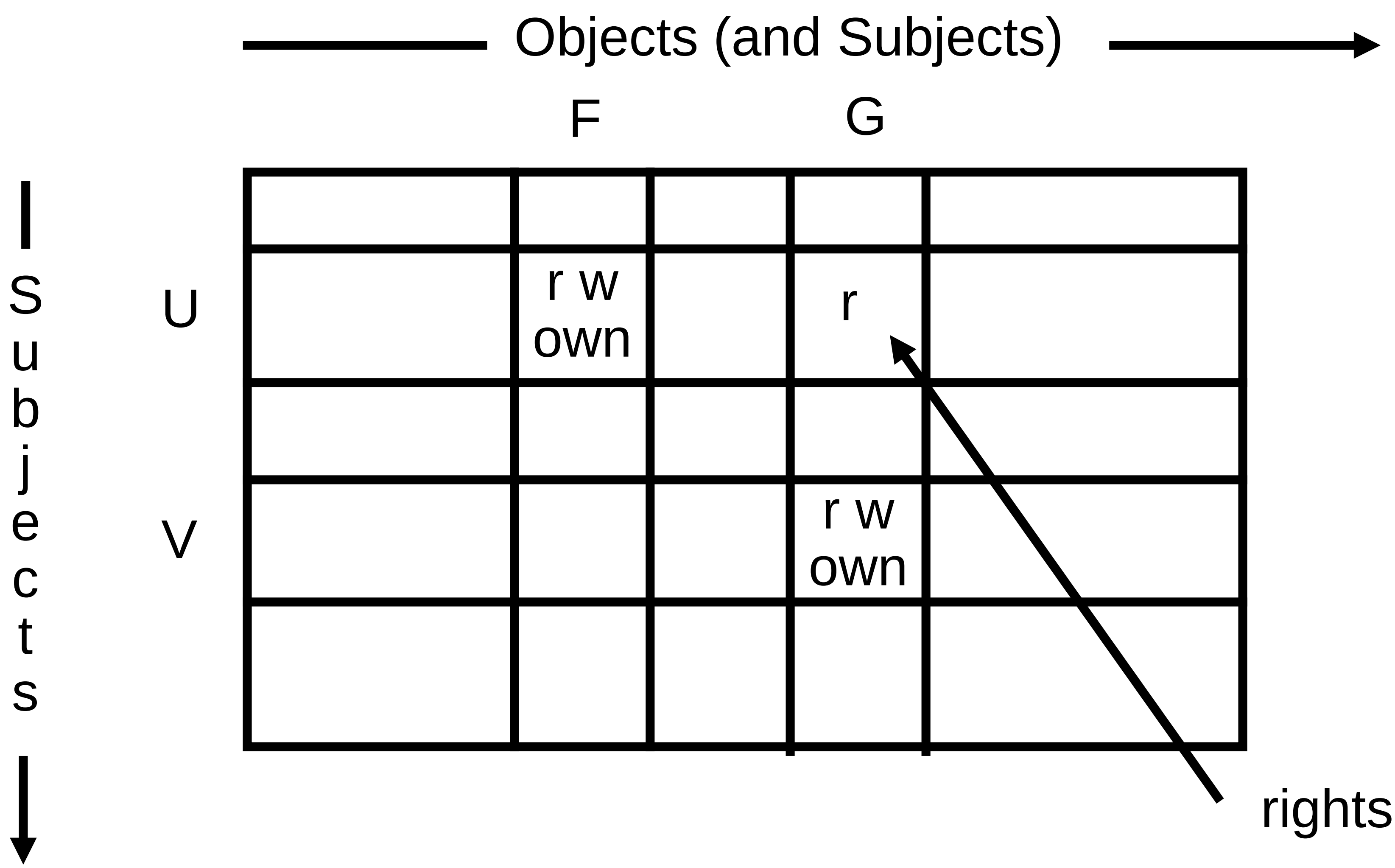
Role Based Access Control (RBAC), 1995

Attribute Based Access Control (ABAC), ????

Flexible policy

Access Matrix Model

Access Matrix Model



Access Matrix Model

- Basic Abstractions
  - Subjects
  - Objects
  - Rights

- The rights in a cell specify the access of the subject (row) to the object (column)

Users and Subjects

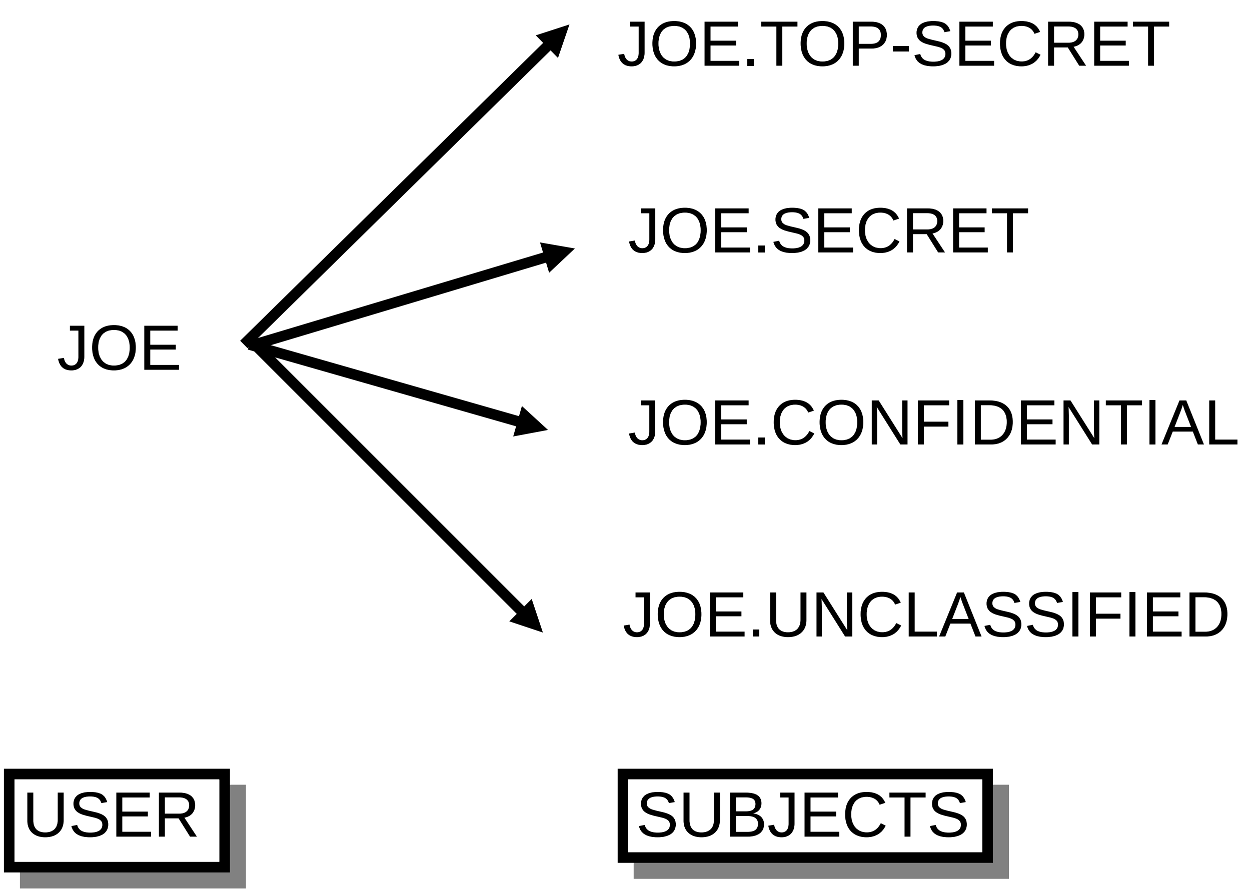
- A subject is a program (application) executing on behalf of a user

- A user may at any time be idle, or have one or more subjects executing on its behalf

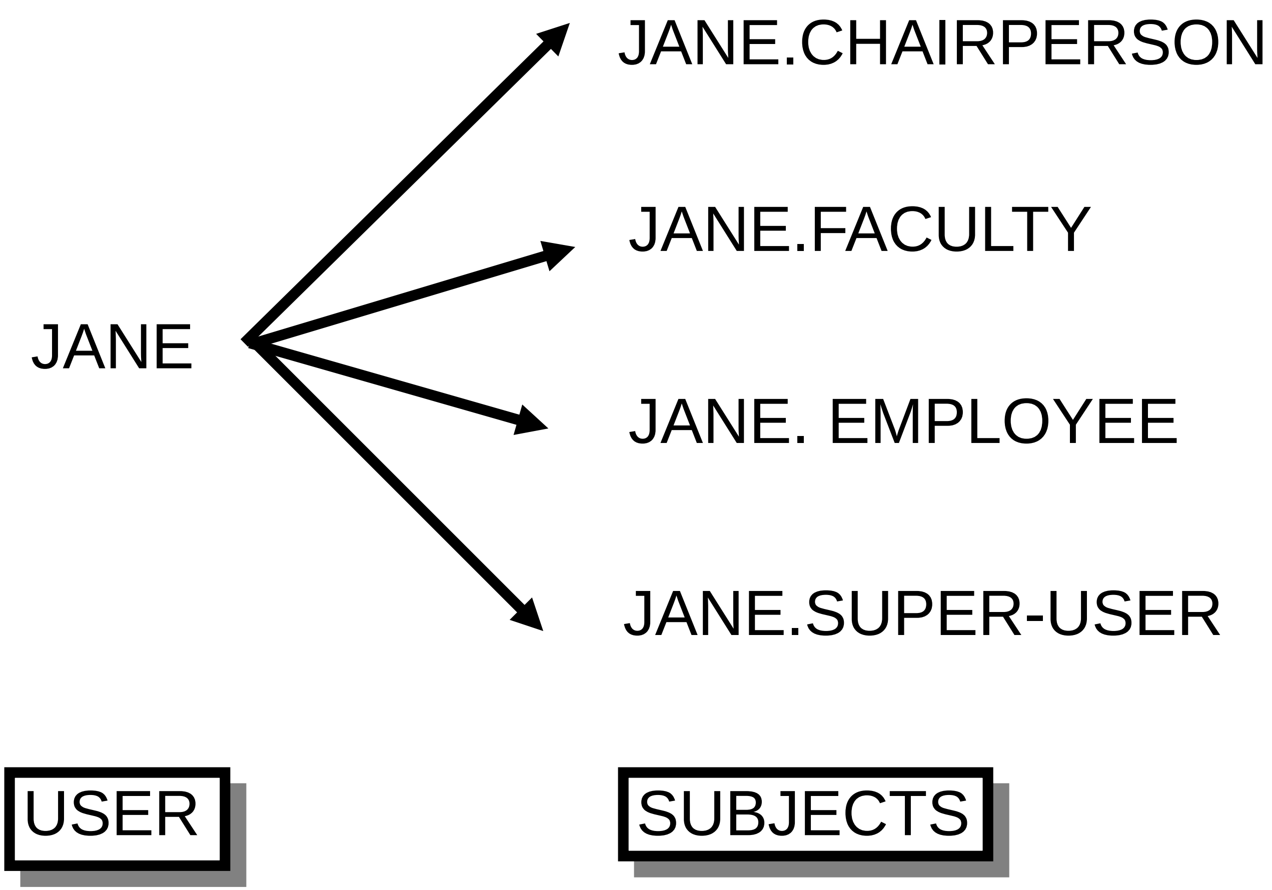
- User-subject distinction is important if subject's rights are different from a user's rights
  - Usually a subset
  - In many systems a subject has all the rights of a user

- A human user may manifest as multiple users (accounts, principals) in the system

Users and Subjects



Users and Subjects



Objects

- An object is anything on which a subject can perform operations (mediated by rights)

- Usually objects are passive, for example:
  - File
  - Directory (or Folder)
  - Memory segment

  with CRUD operations (create, read, update, delete)

- But, subjects can also be objects, with operations
  - kill
  - suspend
  - resume

Access Matrix Model

Objects (and Subjects) →

Subjects ↓

		F		W	
U		r w own		parent	
W		r w own			

Implementation

- Access Control Lists
- Capabilities
- Relations

Access Control Lists

F

U:r
U:w
U:own

G

U:r
V:r
V:w
V:own

each column of the access matrix is stored with the object corresponding to that column

Capabilities

U | F/r, F/w, F/own, G/r |

V | G/r, G/w, G/own |

each row of the access matrix is stored with the subject corresponding to that row

Relations

Subject	Access	Object
U	r	F
U	w	F
U	own	F
U	r	G
V	r	G
V	w	G
V	own	G

commonly used in relational database management systems

ACLs versus Capabilities

- Authentication
  - ACL's require authentication of subjects and ACL integrity
  - Capabilities require integrity and propagation control
- Access review
  - ACL's are superior on a per-object basis
  - Capabilities are superior on a per-subject basis
- Revocation
  - ACL's are superior on a per-object basis
  - Capabilities are superior on a per-subject basis
- Least privilege
  - Capabilities provide for finer grained least privilege control with respect to subjects, especially dynamic short-lived subjects created for specific tasks

ACLs versus Capabilities

- Authentication
  - ACL's require authentication of subjects and ACL integrity
  - Capabilities require integrity and propagation control
- Access review
  - ACL's are superior on a per-object basis
  - Capabilities are superior on a per-subject basis
- Revocation
  - ACL's are superior on a per-object basis
  - Capabilities are superior on a per-subject basis
- Least privilege
  - Capabilities provide for finer grained least privilege control with respect to subjects, especially dynamic short-lived subjects created for specific tasks

Most Operating Systems use ACLs often in abbreviated form: owner, group, world

Content-Dependent Controls

- content dependent controls
  - you can only see salaries less than 50K, or
  - you can only see salaries of employees who report to you

- beyond the scope of Operating Systems and are provided by Database Management Systems

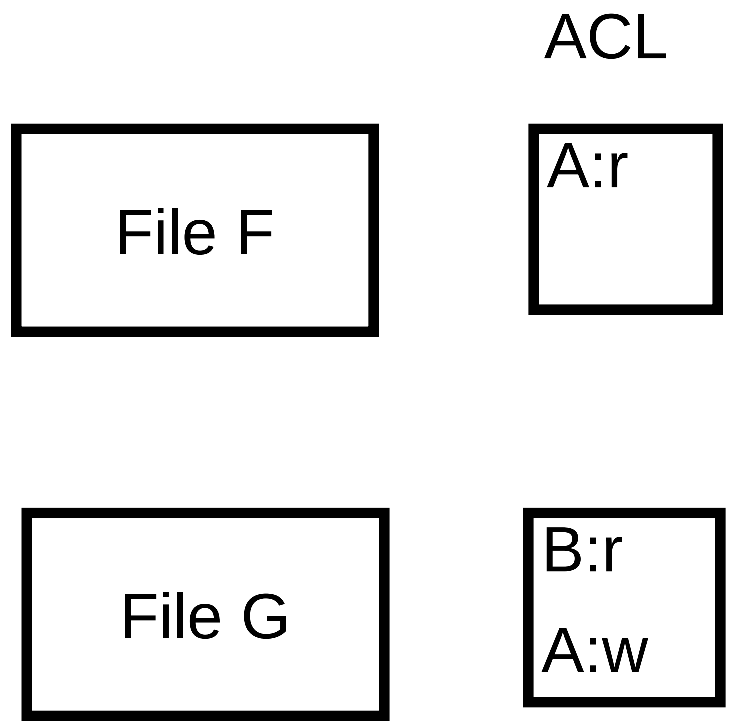
Context-Dependent Controls

- context dependent controls
  - cannot access classified information via remote login
  - salary information can be updated only at year end
  - company's earnings report is confidential until announced at the stockholders meeting
- can be partially provided by the Operating System and partially by the Database Management System
- more sophisticated context dependent controls such as based on past history of accesses definitely require Database support

Trojan Horse Vulnerability of DAC

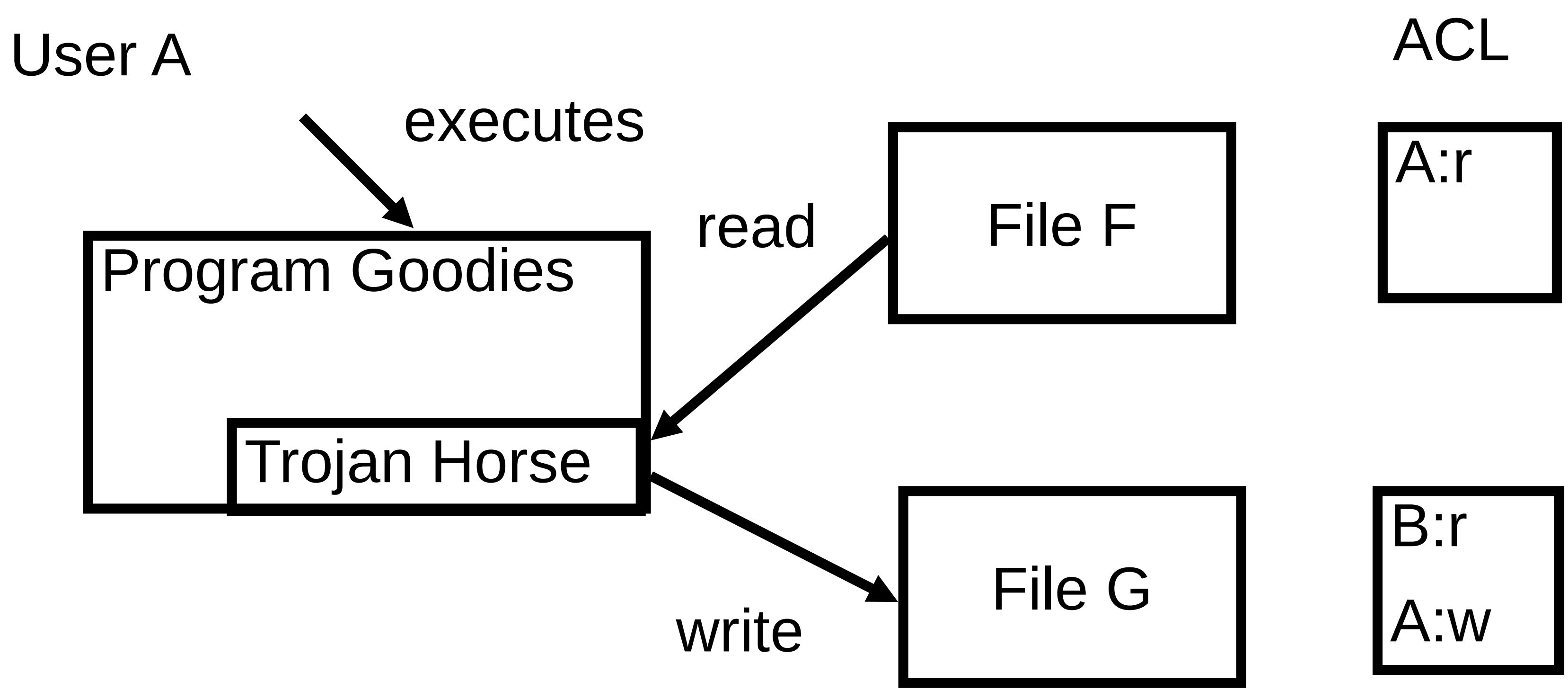
- Information from an object which can be read can be copied to any other object which can be written by a subject
- Suppose our users are trusted not to do this deliberately. It is still possible for Trojan Horses to copy information from one object to another.

Trojan Horse Vulnerability of DAC



User B cannot read file F

Trojan Horse Vulnerability of DAC



User B can read contents of file F copied to file G

Copy Difference for rw

- Read of a digital copy is as good as read of original
- Write to a digital copy is not so useful

DAC Subtleties

- Chains of grants and revokes
- Inheritance of permissions
- Negative rights

Denning's Axioms for Information Flow

Denning's Axioms

$$< SC, \rightarrow, \oplus >$$

SC	set of security classes
$\rightarrow \subseteq SC \times SC$	flow relation (i.e., can-flow)
$\oplus$: SC X SC -> SC	class-combining operator

Denning's Axioms

$$< SC, \rightarrow, \oplus >$$

1. SC is finite
2. $\rightarrow$ is a partial order on SC
   (i.e., reflexive, transitive, anti-symmetric)
3. SC has a lower bound L such that L $\rightarrow$ A for all A $\in$ SC
4. $\oplus$ is a least upper bound (lub) operator on SC

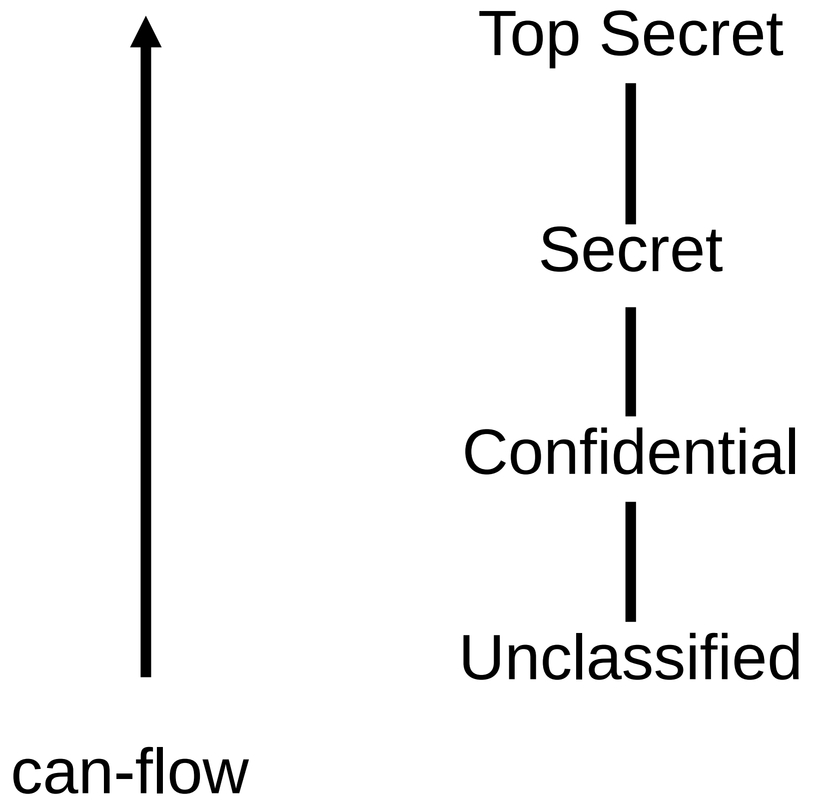
Justification for 1 and 2 is stronger than for 3 and 4.
In practice we may have a partially ordered set (poset).

Denning's Axioms Imply

- SC is a universally bounded lattice
- There exists a Greatest Lower Bound (glb) operator ⊗ (also called meet)
- There exists a highest security class H

Lattice Structures

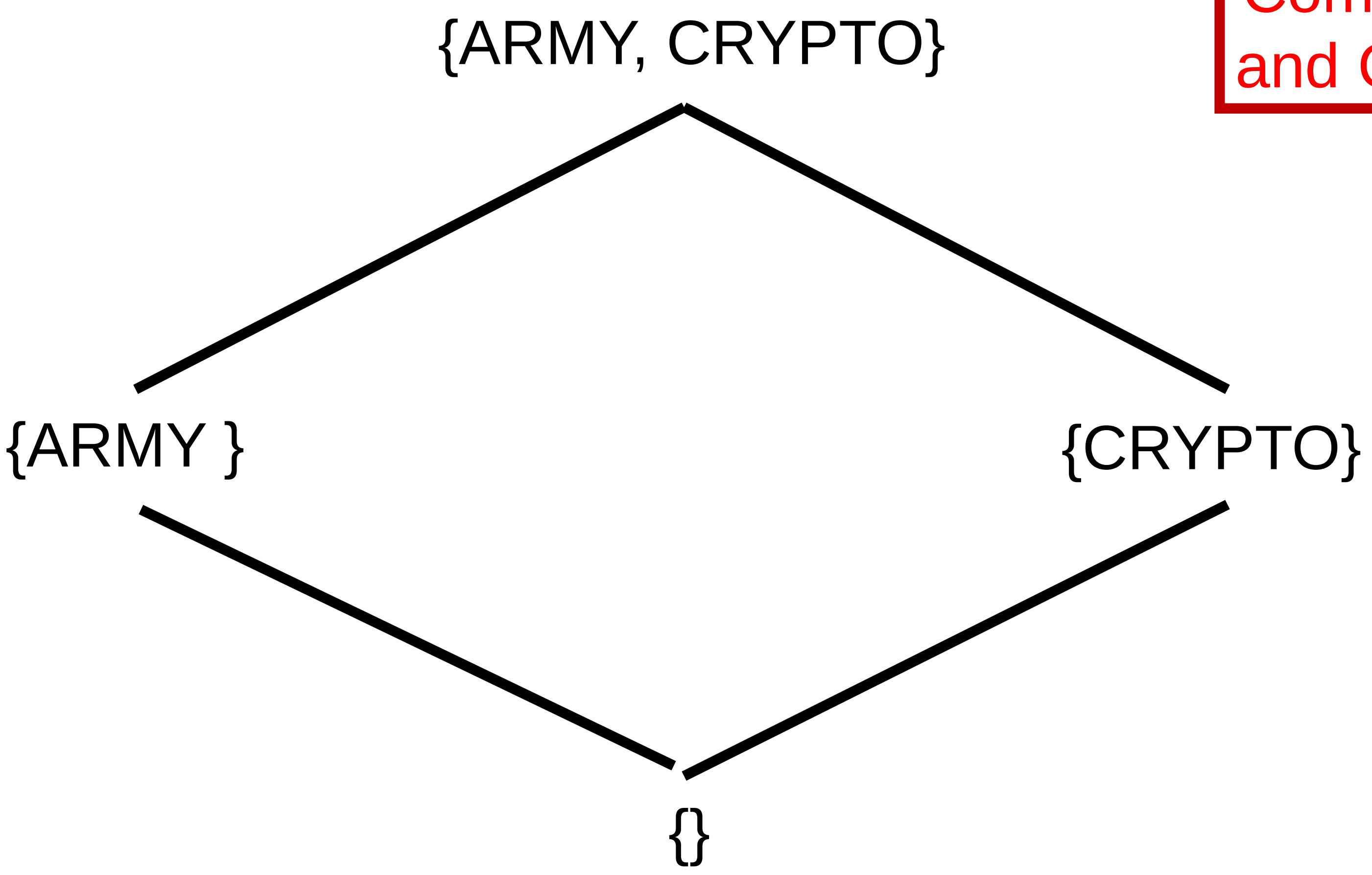
Hierarchical Classes



reflexive and transitive edges are implied but not shown

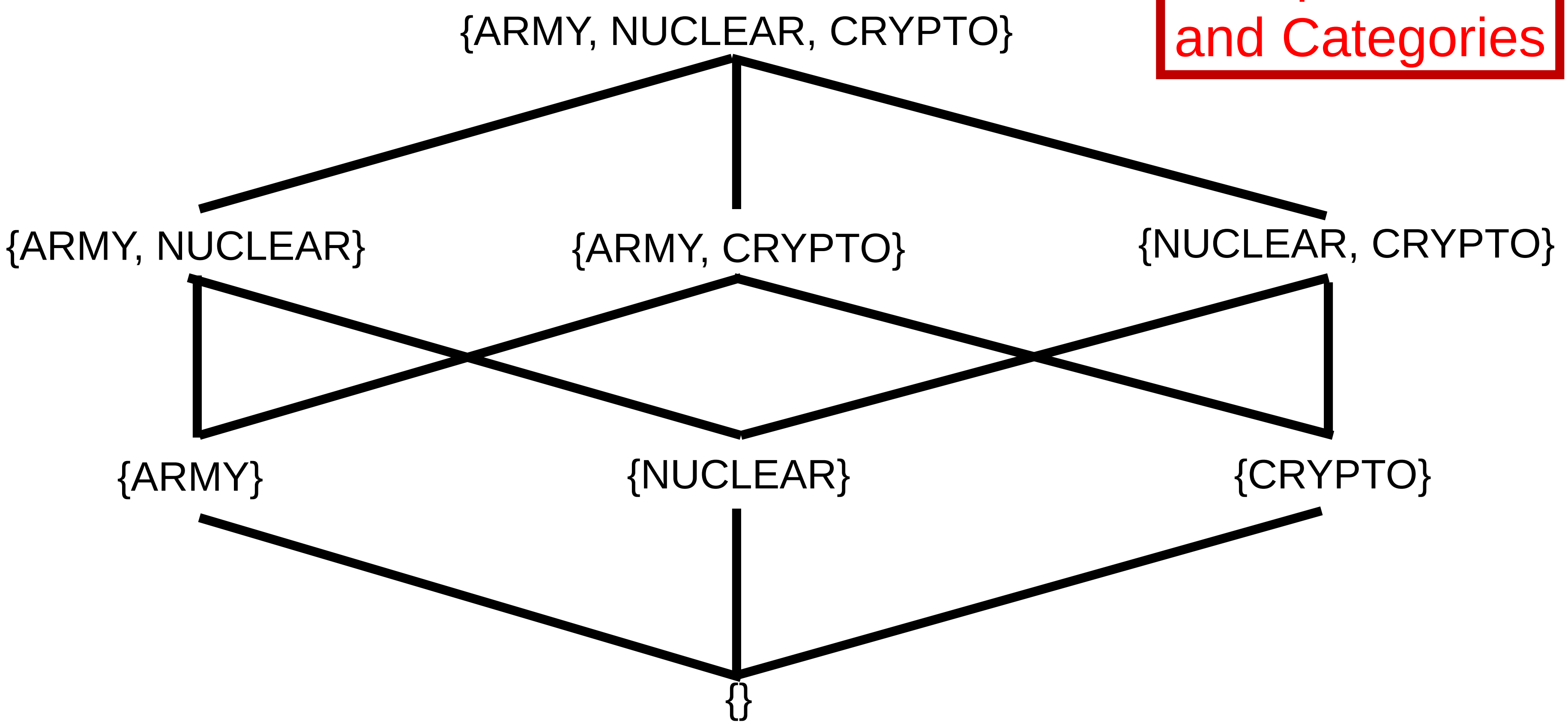
Lattice Structures

Compartments and Categories

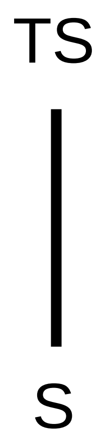


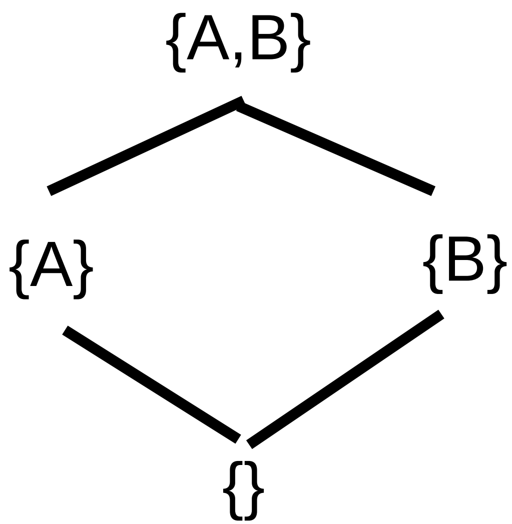
Lattice Structures

Compartments and Categories



Lattice Structures

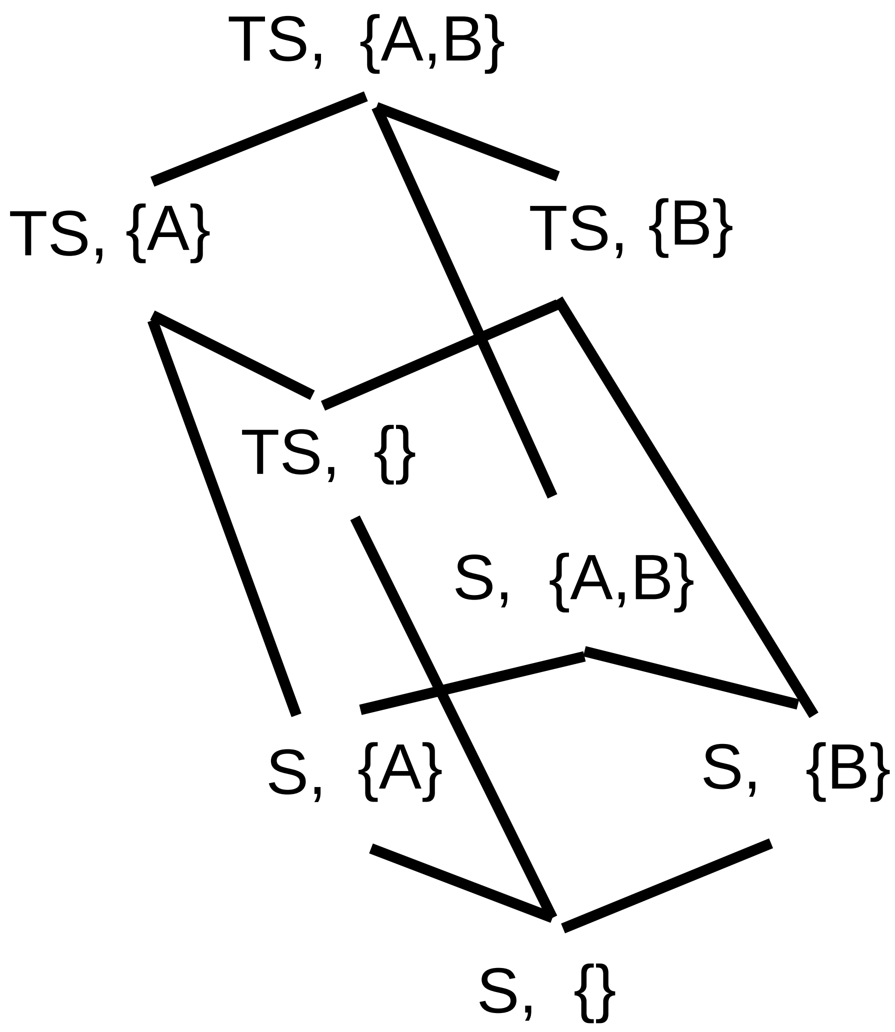




Hierarchical Classes with Compartments

product of 2 lattices is a lattice

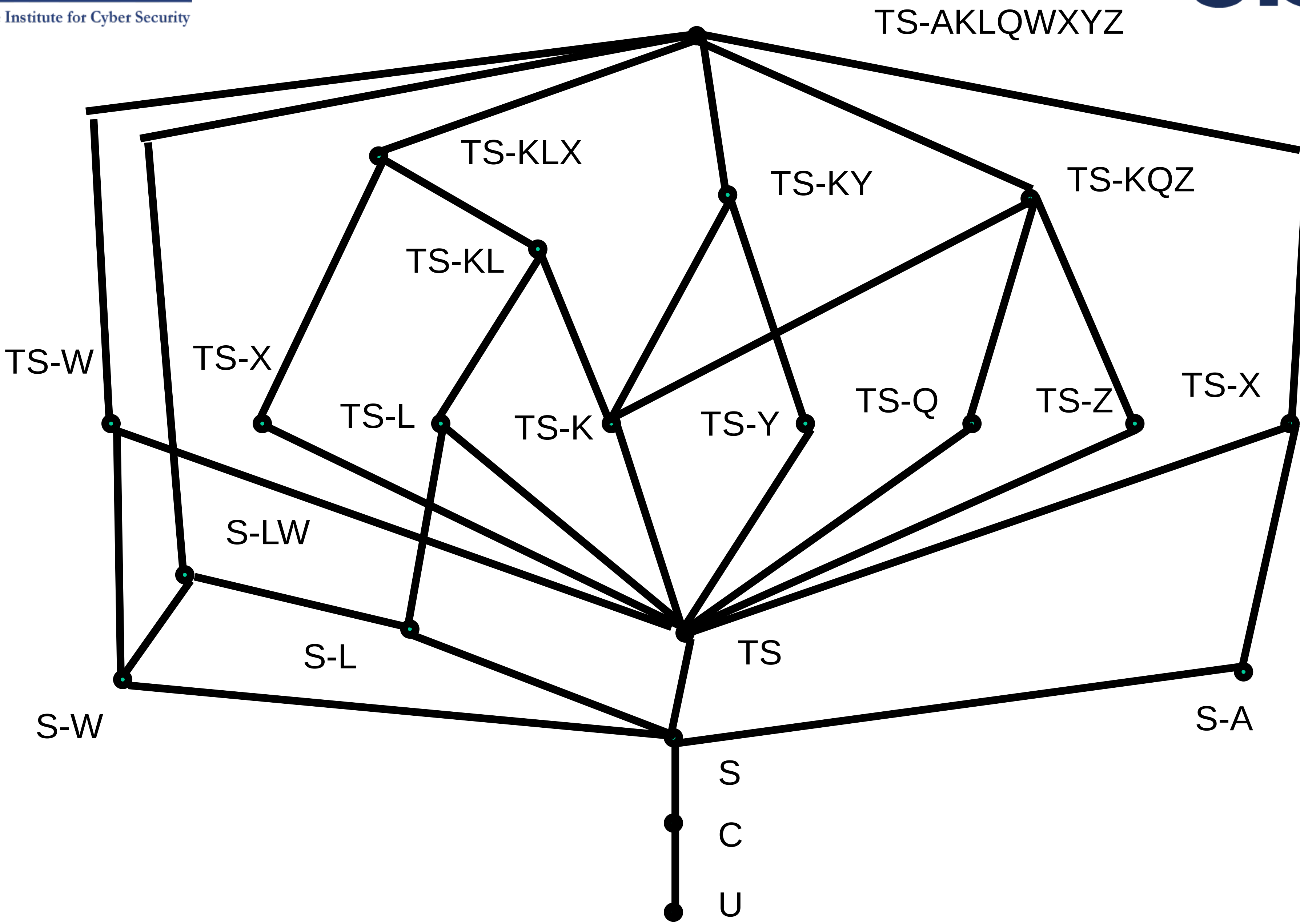
Lattice Structures



Hierarchical Classes with Compartments

product of 2 lattices is a lattice

Smith's Lattice

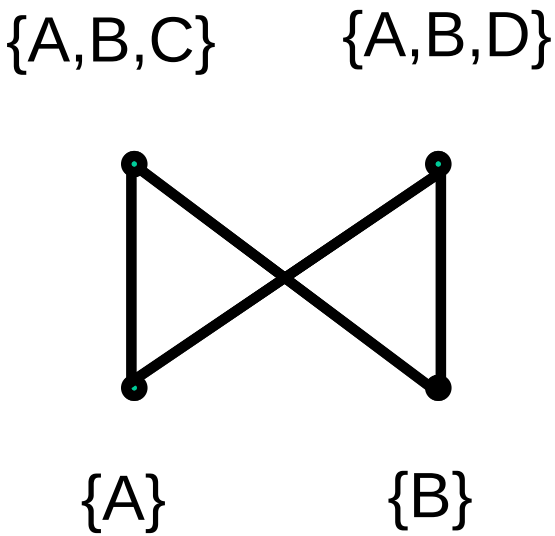


World-Leading Research with Real-World Impact!

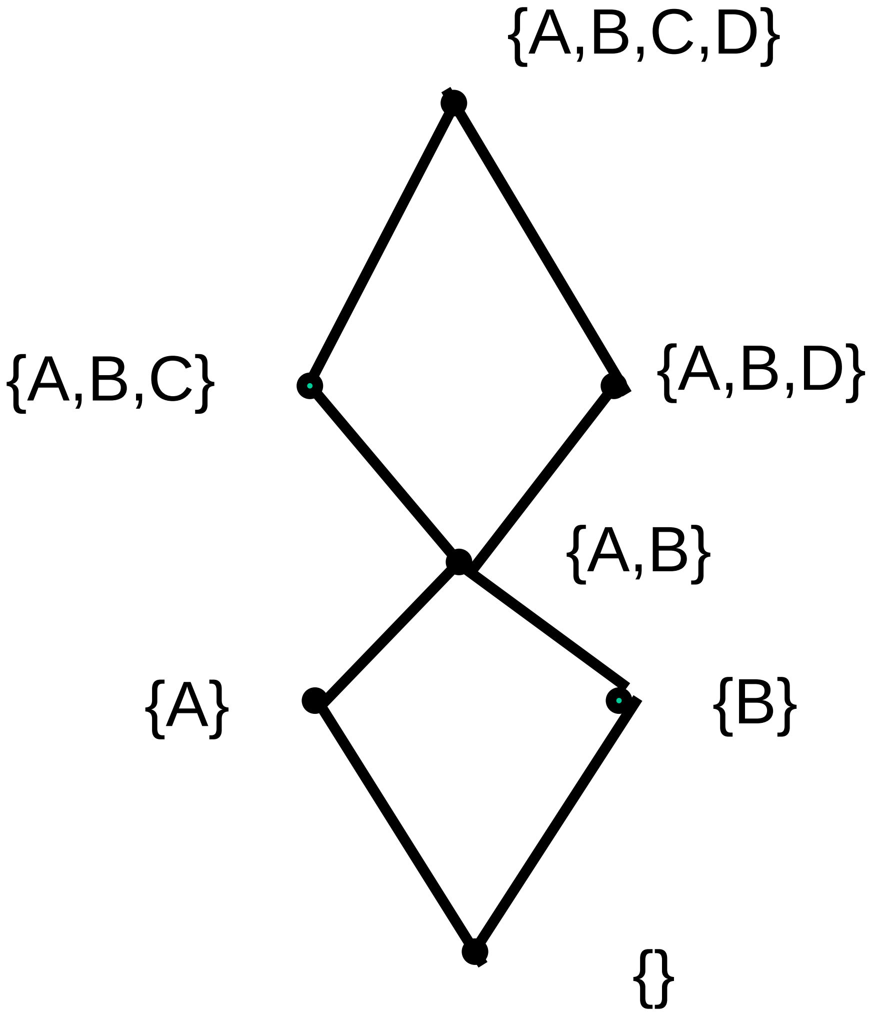
Smith's Lattice

- With large lattices a vanishingly small fraction of the labels will actually be used
  - Smith's lattice: 4 hierarchical levels, 8 compartments
  - number of possible labels = 4*2^8 = 1024
    Only 21 labels are actually used (2%)
- Consider 16 hierarchical levels, 64 compartments which gives 10^20 labels

Extending a POSET to a Lattice



⇒



such extension
is always possible

BLP Model for Confidentiality

BLP Basic Assumptions

- SUB = {S1, S2, ..., Sm}, a fixed set of subjects
- OBJ = {O1, O2, ..., On}, a fixed set of objects
- R = {r, w}, a fixed set of rights
- D, an $m \times n$ discretionary access matrix with $D[i,j] \subseteq R$
- M, an $m \times n$ current access matrix with $M[i,j] \subseteq R$

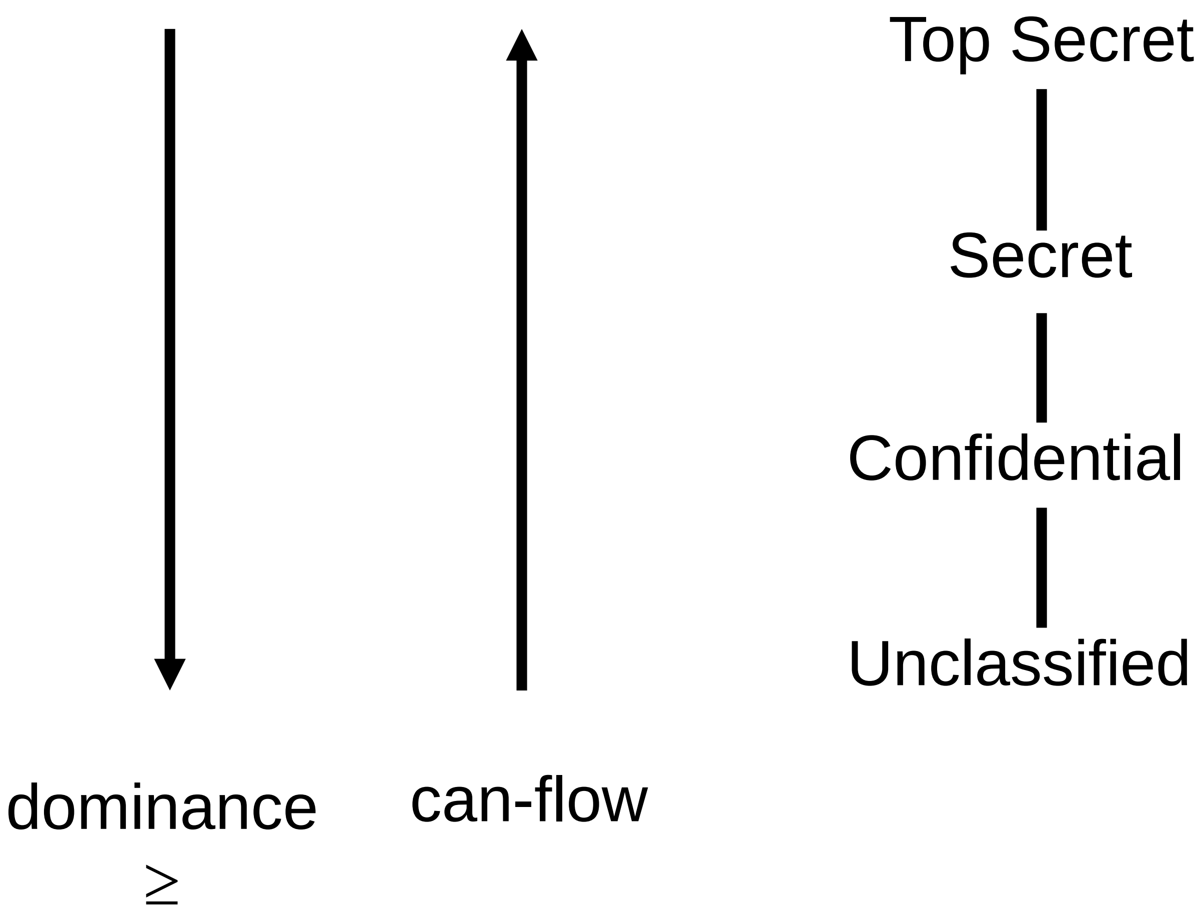
BLP Model (Liberal ★-Property)

- Lattice of confidentiality labels $\Lambda = \{\lambda 1, \lambda 2, \ldots, \lambda p\}$
- Static assignment of confidentiality labels λ: SUB $\cup$ OBJ $\rightarrow \Lambda$
- M, an m $\times$ n current access matrix with
  - r $\in$ M[i,j] $\Rightarrow$ r $\in$ D[i,j] $\wedge$ λ(Si) $\geq$ λ (Oj) simple security
  - w $\in$ M[i,j] $\Rightarrow$ w $\in$ D[i,j] $\wedge$ λ(Si) $\leq$ λ (Oj) liberal ★-property

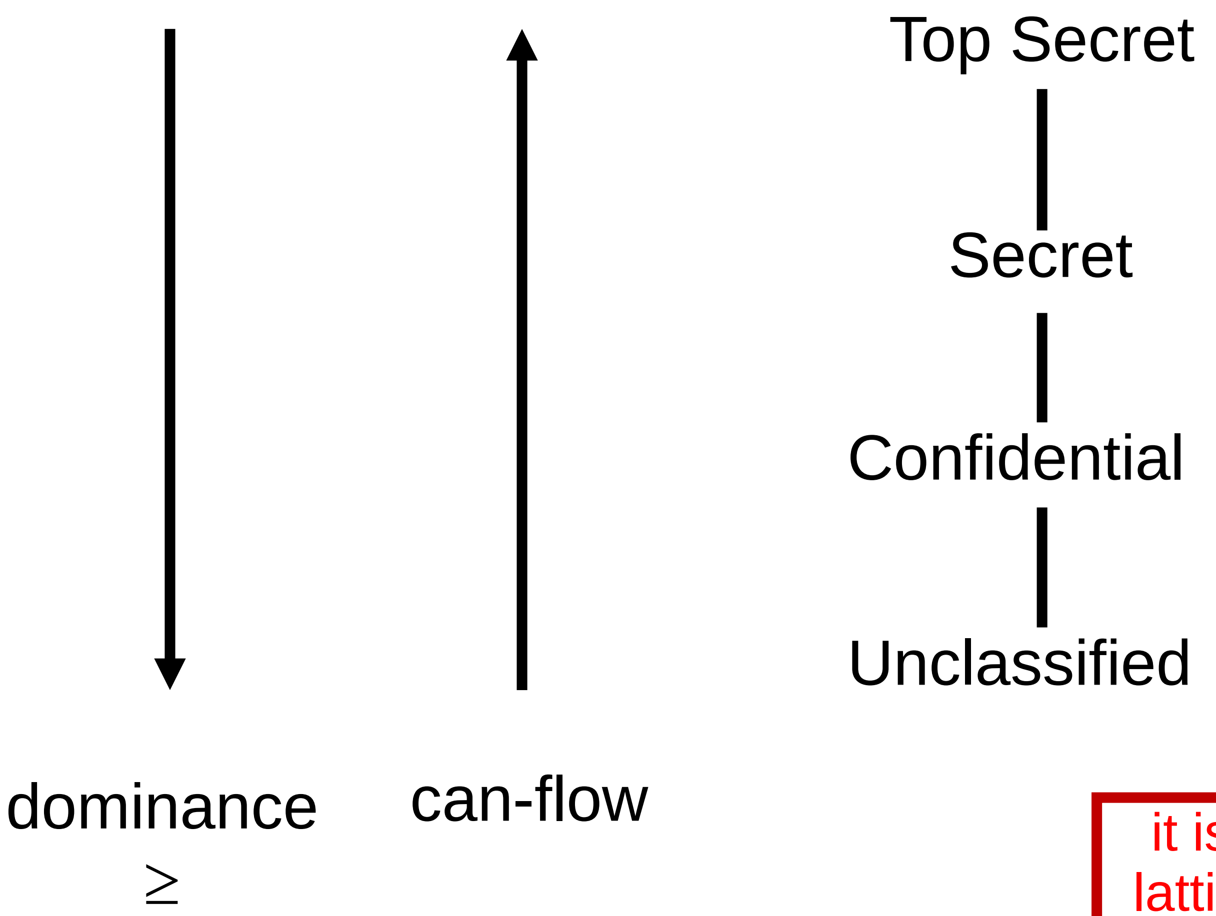
BLP Model (Strict ★-Property)

- Lattice of confidentiality labels $\Lambda = \{\lambda 1, \lambda 2, ..., \lambda p\}$
- Static assignment of confidentiality labels λ: SUB $\cup$ OBJ $\rightarrow \Lambda$
- M, an m $\times$ n current access matrix with
  - r $\in$ M[i,j] $\Rightarrow$ r $\in$ D[i,j] $\wedge$ λ(Si) $\geq$ λ (Oj) simple security
  - w $\in$ M[i,j] $\Rightarrow$ w $\in$ D[i,j] $\wedge$ λ(Si) $=$ λ (Oj) strict ★-property

BLP vis a vis Lattices

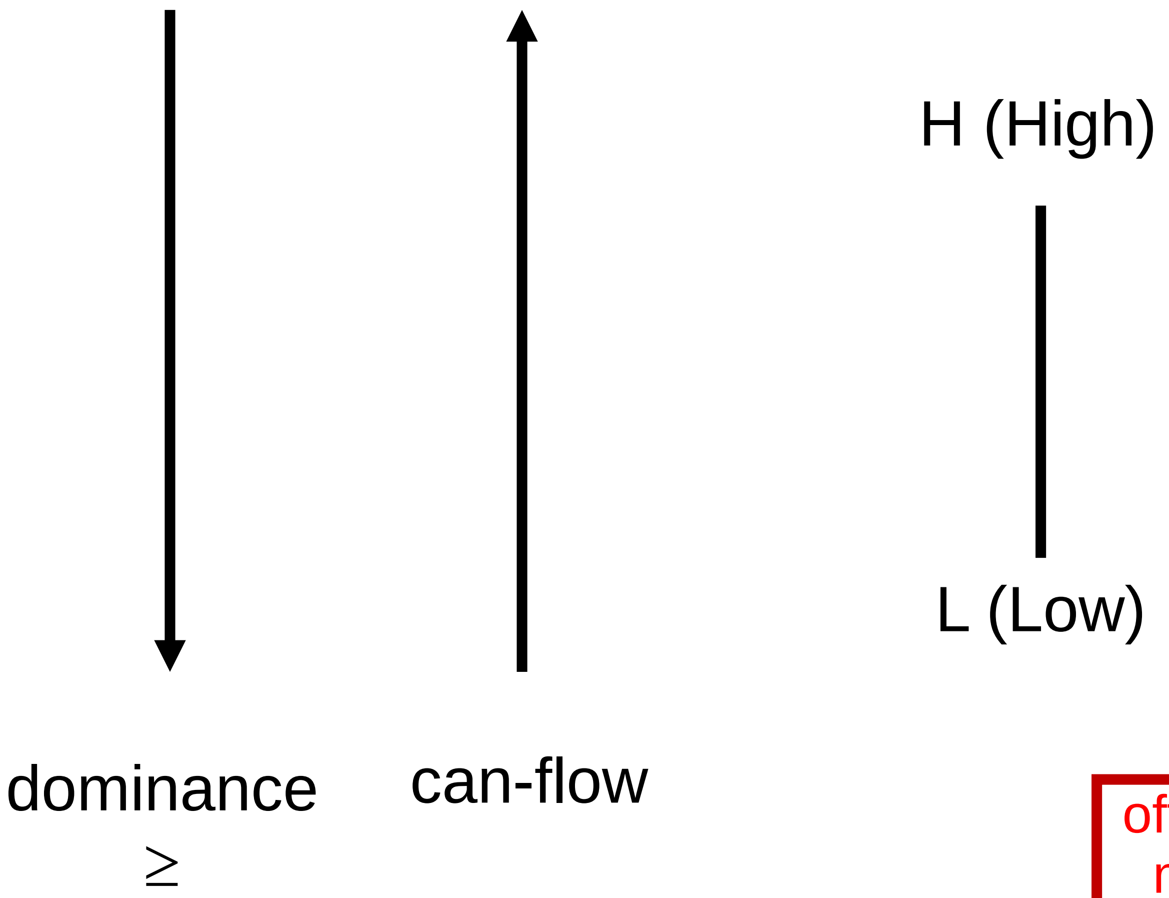


BLP vis a vis Lattices



it is risky to visualize lattices as total orders but it is ok sometimes

BLP vis a vis Lattices



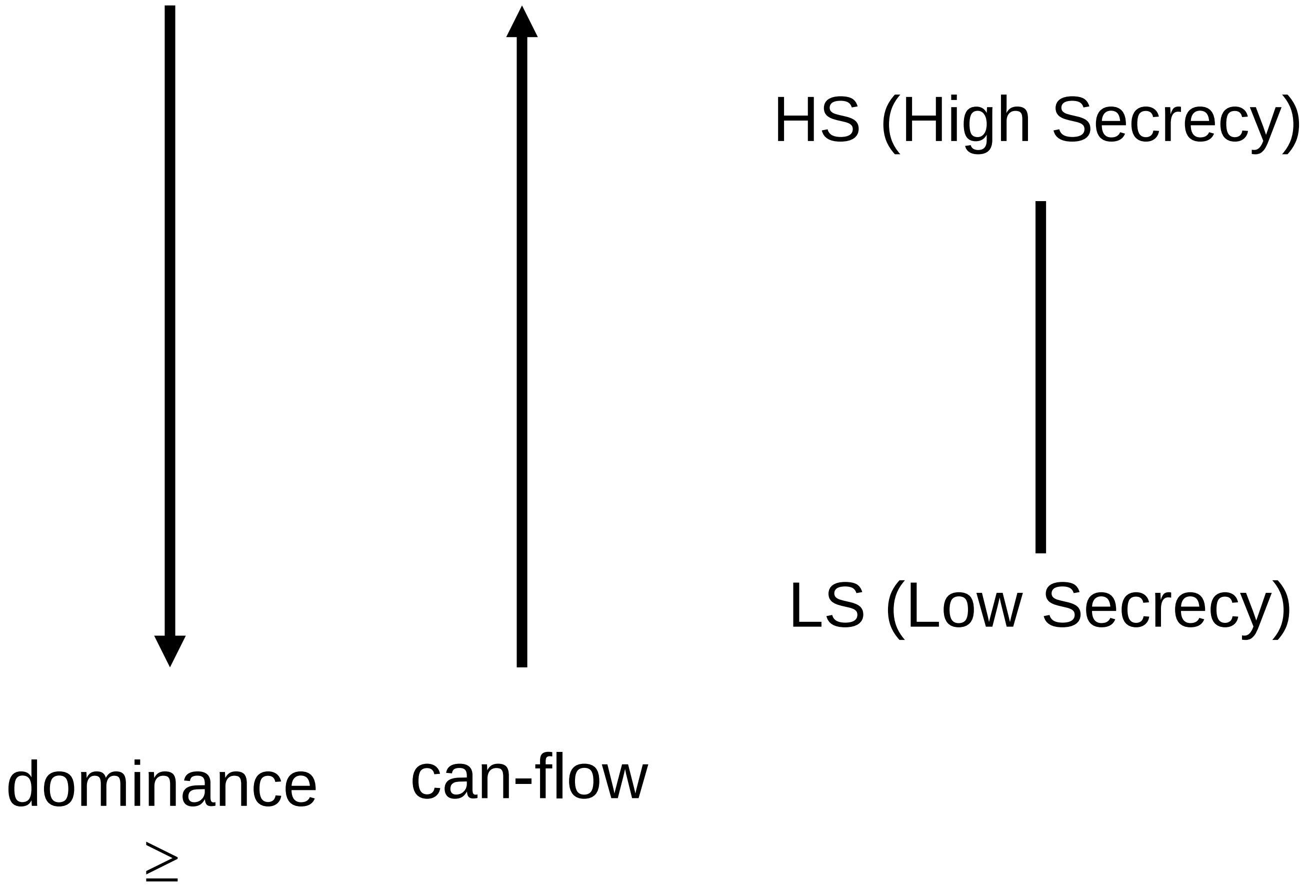
often 2 levels suffice to make the main point

★-Property

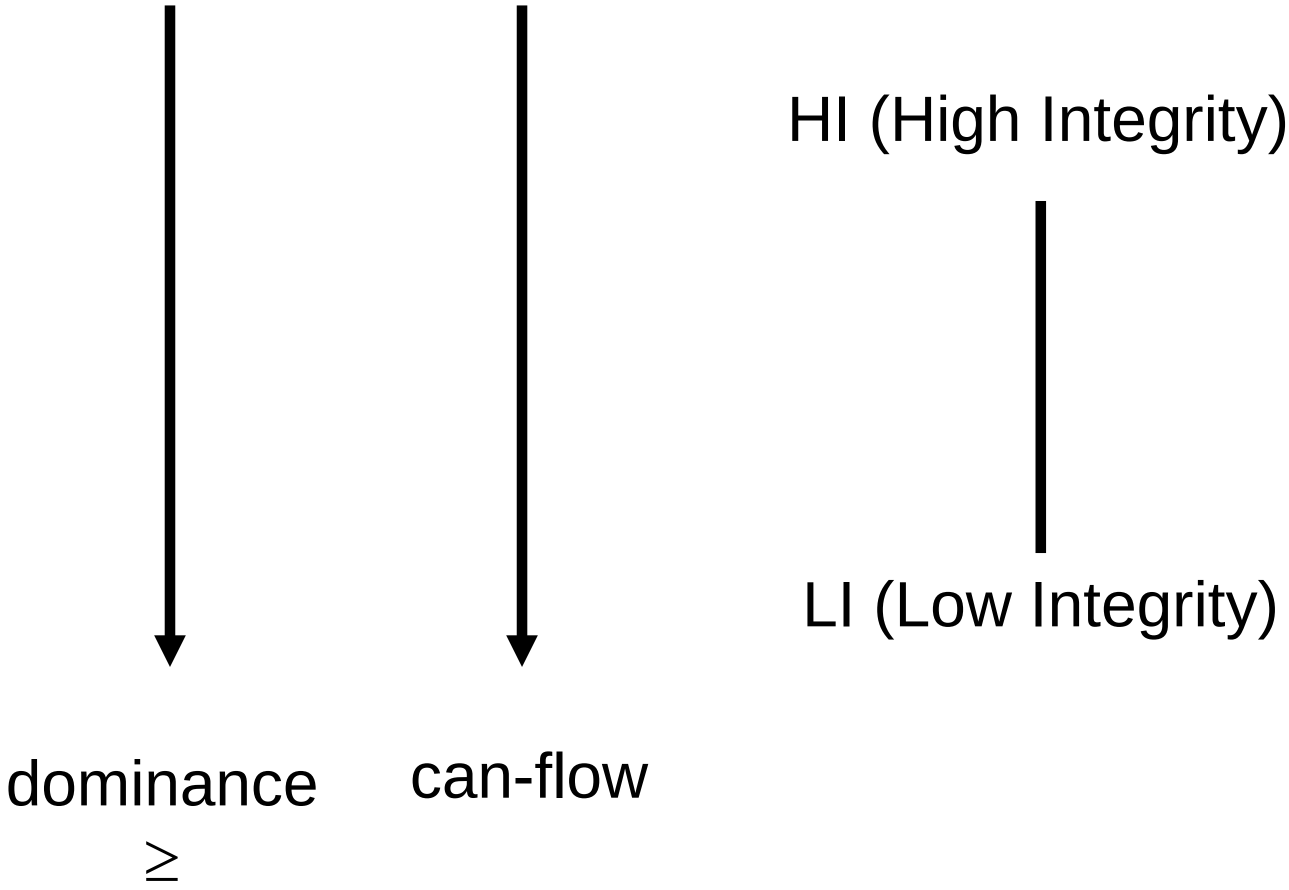
- Applies to subjects not to users
  - Users are trusted (must be trusted) not to disclose secret information outside of the computer system
  - A user can login (create a subject) with any label dominated by the user's clearance
  - Subjects are not trusted because they may have Trojan Horses embedded in the code they execute
- ★-property prevents deliberate leakage and does not address
  - inference
  - covert channels
- Simple-security and ★-Property do not account for
  - encryption

Biba Model
for
Integrity

BLP Revisited



Biba Inverted Flow



Biba and BLP Aligned: BLP Style

HS (High Secrecy)

LS (Low Secrecy)

LI (Low Integrity)

HI (High Integrity)

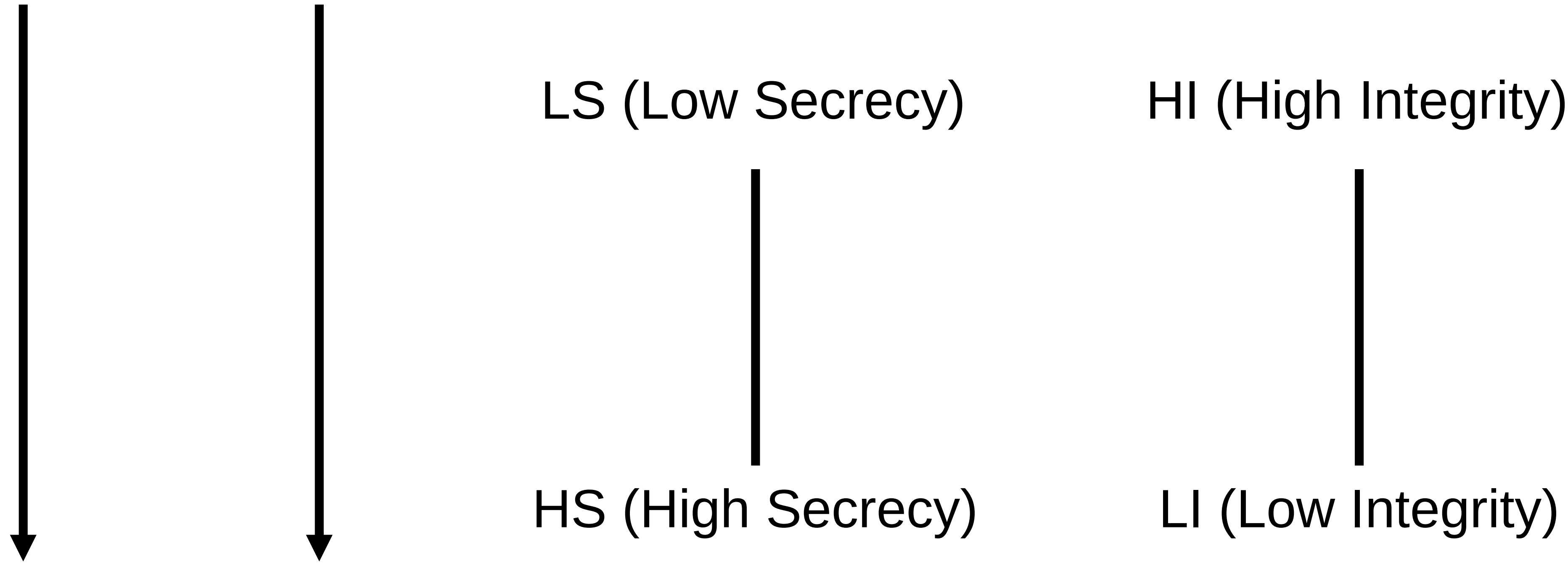
dominance

$\geq$

can-flow

One-directional flow is the key point
No need for opposite directions for confidentiality and integrity

Biba and BLP Aligned: Biba Style

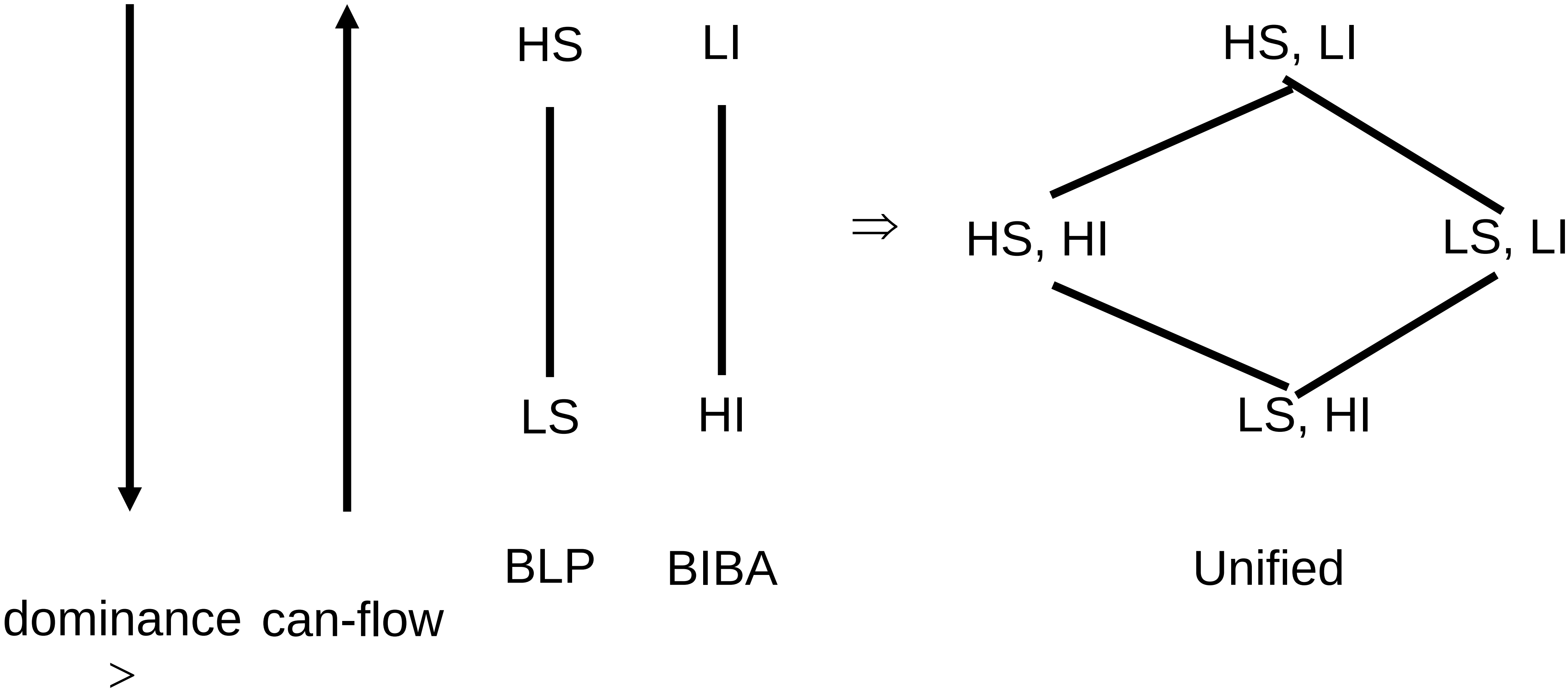


dominance
≥

can-flow

One-directional flow is the key point
No need for opposite directions for confidentiality and integrity

BLP-Biba Unified Lattice: BLP Style



BLP versus Biba

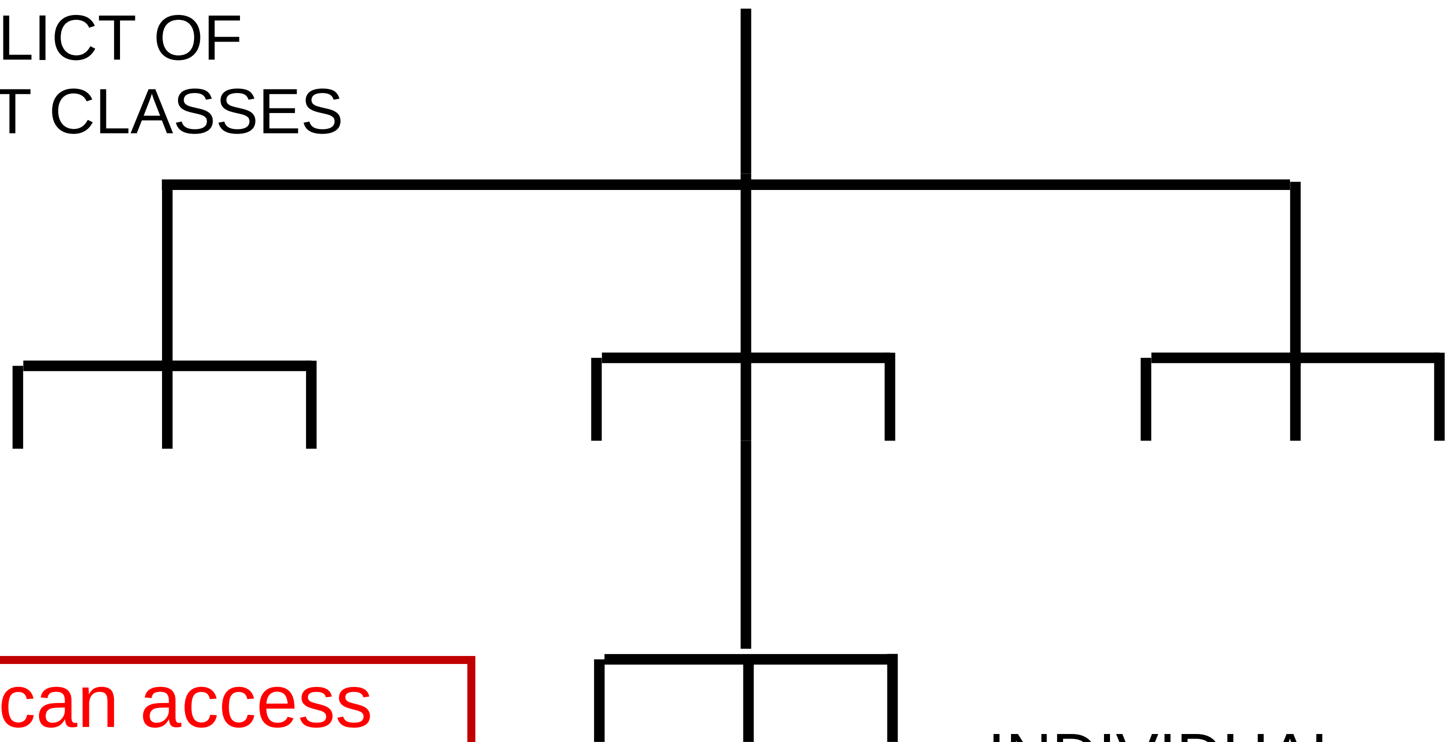
- BLP and Biba are fundamentally equivalent and interchangeable
- Lattice-based access control is a mechanism for enforcing one-way information flow, which can be applied to confidentiality or integrity goals
- We will use the BLP formulation:
  - high confidentiality, low integrity at the top
  - low confidentiality, high integrity at the bottom

The Chinese Wall Lattice for Separation of Duty

Chinese Wall Policy

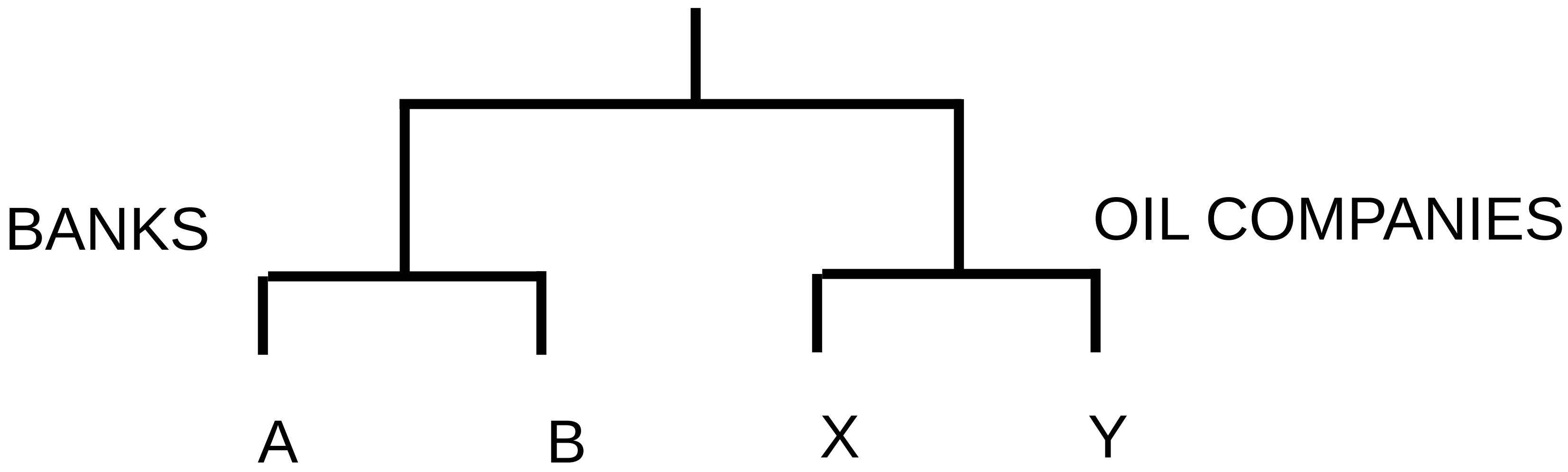
- A commercial security policy for separation of duty driven confidentiality
- Mixture of free choice (discretionary) and mandatory controls
- Requires some kind of dynamic labelling

Chinese Wall Policy

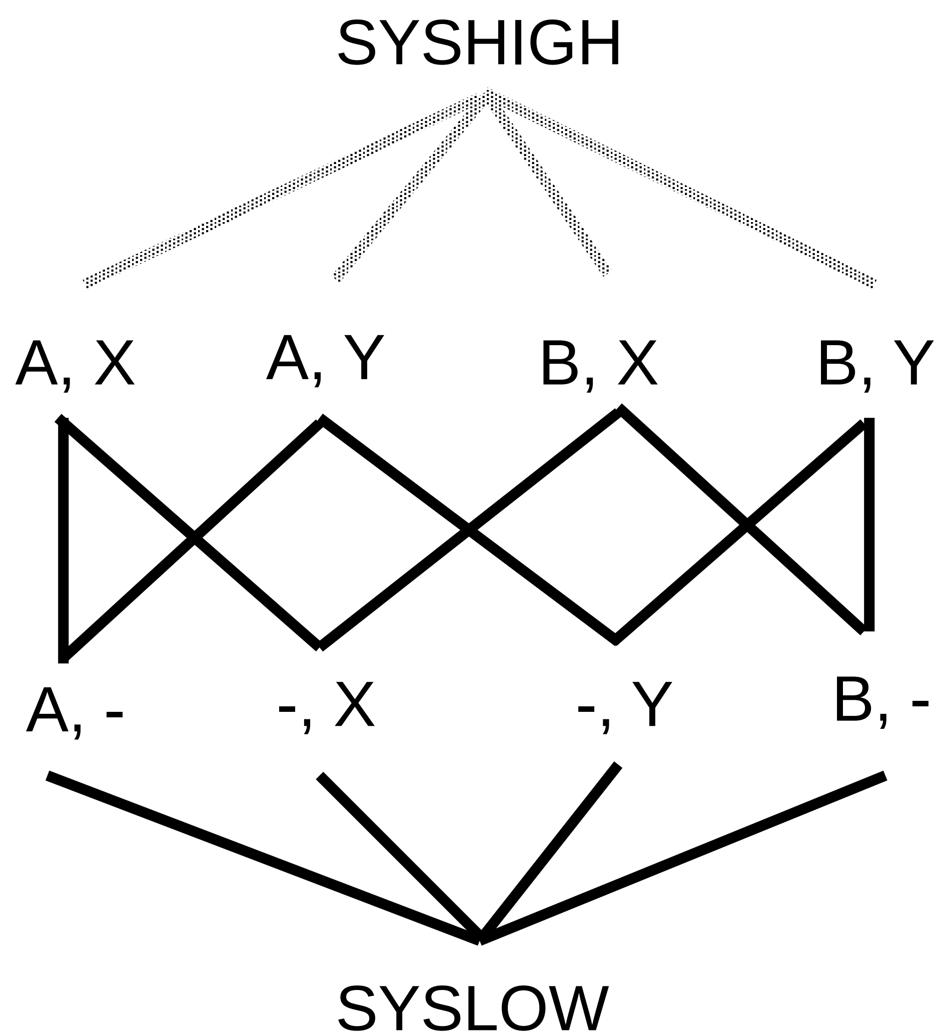


A consultant can access information about at most one company in each conflict of interest class

Chinese Wall Example



Chinese Wall Lattice



Conclusion

MAC or LBAC or BLP (or Biba)

- BLP enforces one-directional information flow in a lattice of security labels

Enforcement

- BLP can enforce one-directional information flow policies for
  - Confidentiality
  - Integrity
  - Separation of duty
  - Combinations thereof

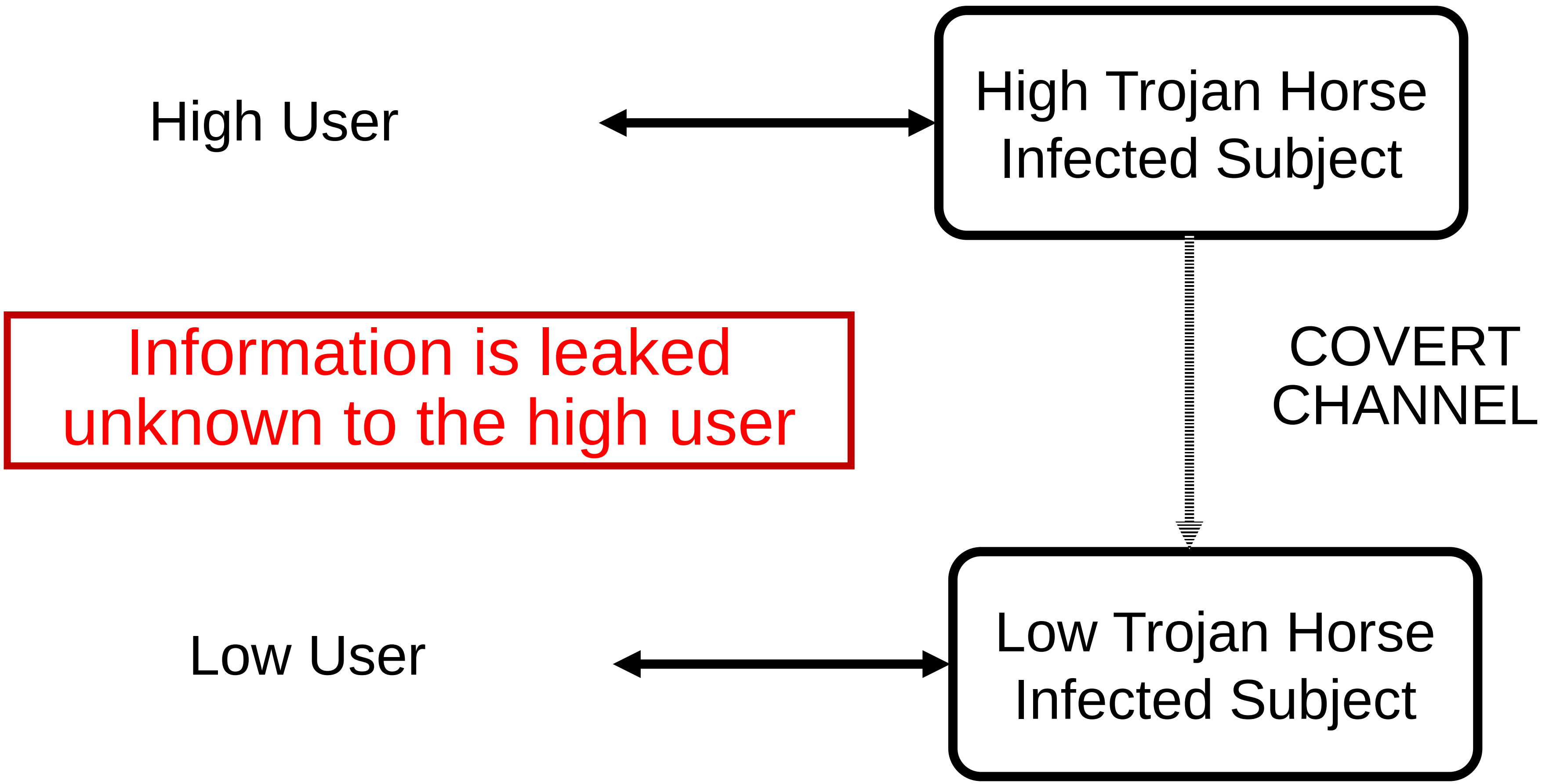
Policy

Covert Channels

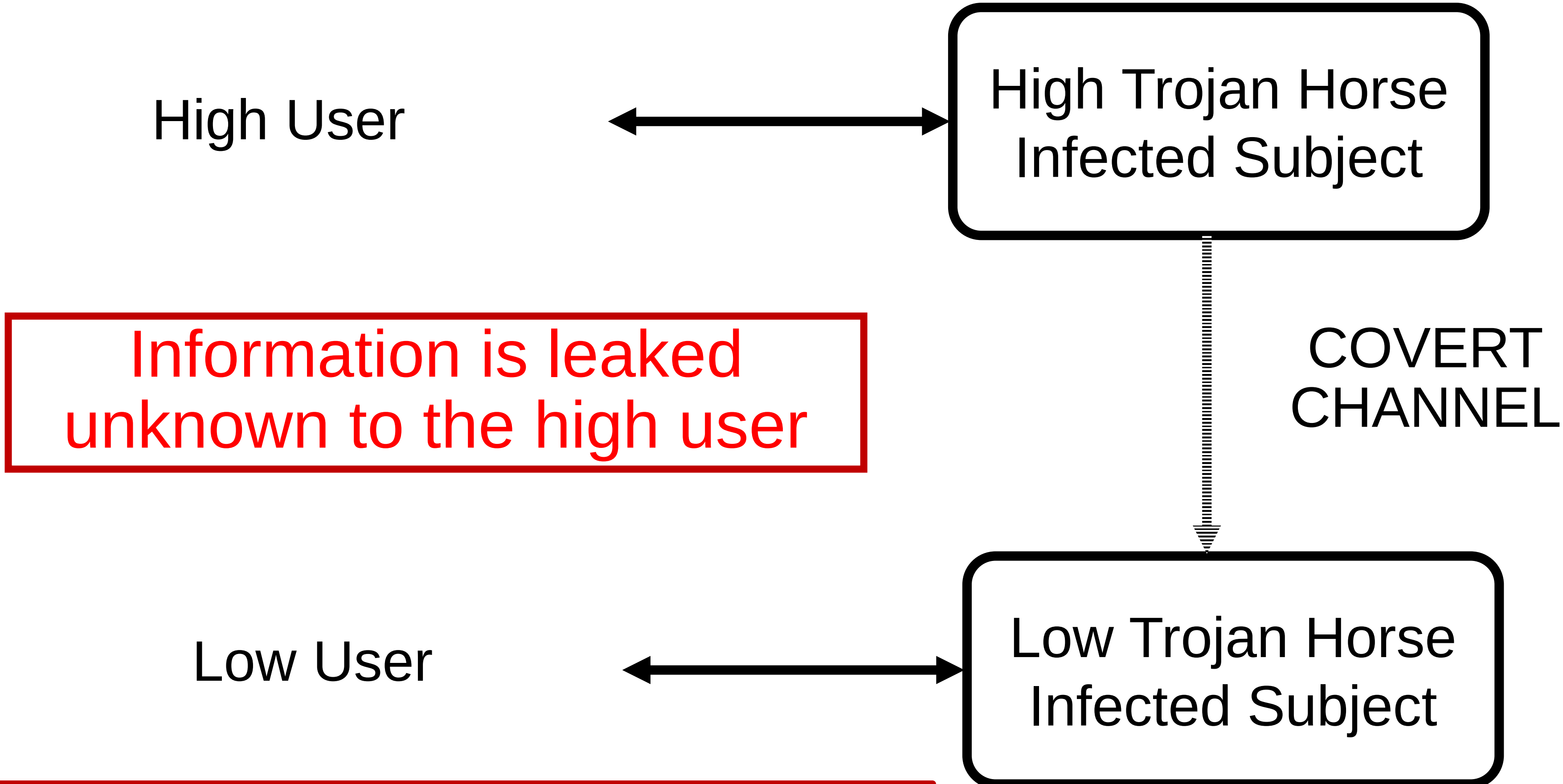
Covert Channels

- A covert channel is a communication channel based on the use of system resources not normally intended for communication between subjects (processes)

Covert Channels

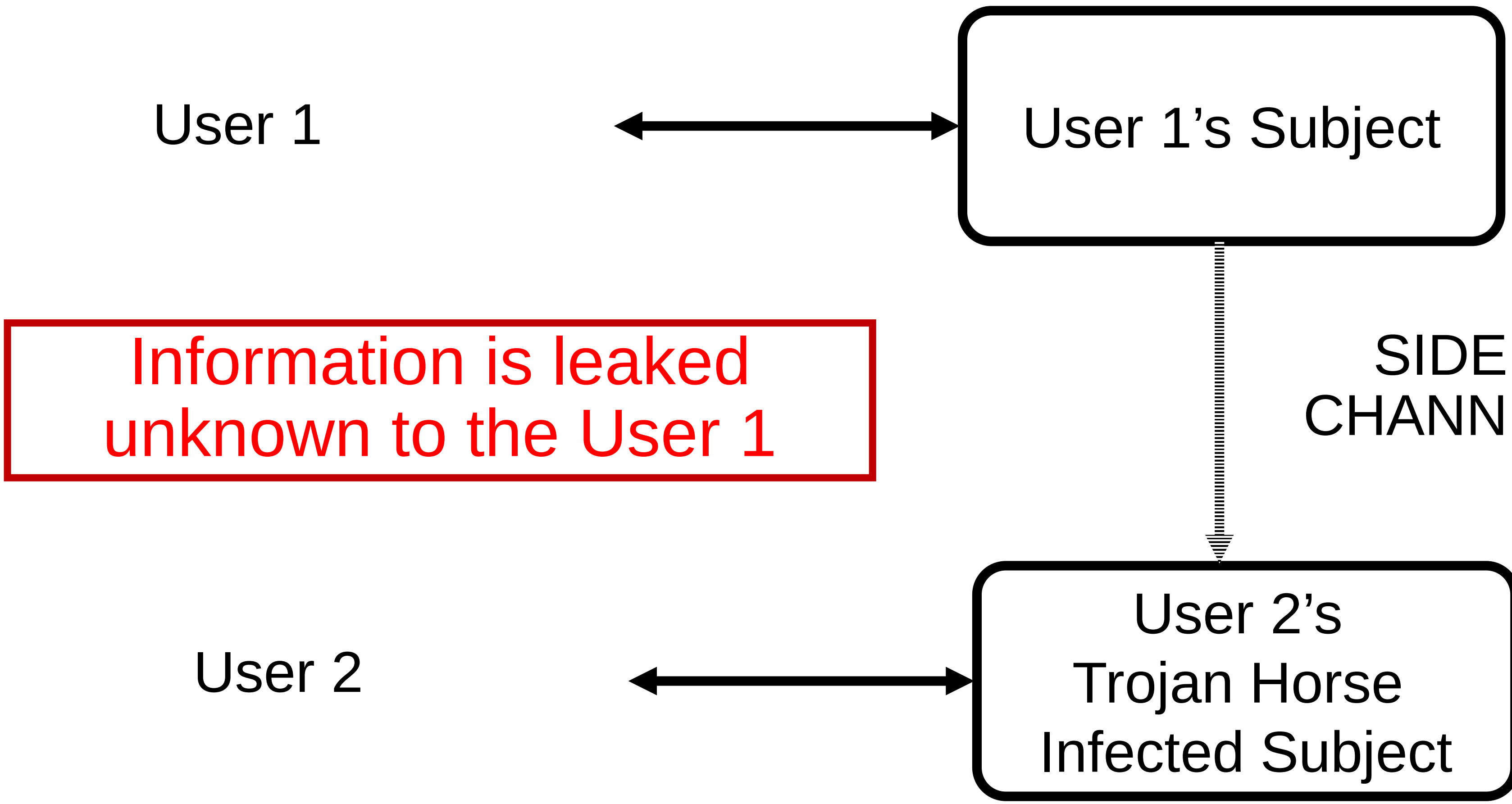


Covert Channels



★-property prevents overt leakage of information and does not address covert channels

Side Channels



Covert Channels versus Side Channels

- Covert channels require a cooperating sender and receiver
- Side channels do not require a sender but nevertheless information is leaked to a receiver

Coping with Covert/Side Channels

- Identify the channel
  - close the channel or slow it down
  - detect attempts to use the channel
  - tolerate its existence

Storage Channels

- Also known as Resource Exhaustion Channels
- Given 5GB pool of dynamically allocated memory
  - HIGH PROCESS (sender)
    bit = 1 $\Rightarrow$ request 5GB of memory
    bit = 0 $\Rightarrow$ request 0GB of memory
  - LOW PROCESS (receiver)
    request 5GB of memory
    if allocated then bit = 0 otherwise bit = 1

Timing Channels

- Also known as Load Sensing Channels
- Given 5GB pool of dynamically allocated memory
  - HIGH PROCESS (sender)
    bit = 1 $\Rightarrow$ enter computation intensive loop
    bit = 0 $\Rightarrow$ go to sleep
  - LOW PROCESS (receiver)
    perform a task with known computational requirement
    if completed promptly then bit = 0 otherwise bit = 1