Mandatory Policies

Towards Mandatory Policies

- Processes run programs which, unless properly certified, cannot be trusted for the operations they execute
- For this reason, restrictions should be enforced on the operations that processes themselves can execute
- In particular, protection against Trojan Horses leaking information to unauthorized users requires controlling the flows of information within processes execution and possibly restricting them
- Mandatory policies provide a way to enforce information flow control through the use of labels

Mandatory Flow Control Models

- Definition: Mandatory access control refers to a type of access control by which the operating system constrains the ability of a subject or initiator to access or generally perform some sort of operation on an object or target.

- Why is it necessary since we have discretionary security model?

- With the advances in networks and distributed systems, it is necessary to broaden the scope to include the control of information flow between distributed nodes on a system wide basis rather than only individual basis like discretionary control.

Difference between Discretionary and Mandatory access control

- Mandatory access control, this security policy is centrally controlled by a security policy administrator; users do not have the ability to override the policy and, for example, grant access to files that would otherwise be restricted.

- By contrast, discretionary access control (DAC), which also governs the ability of subjects to access objects, allows users the ability to make policy decisions and/or assign security attributes.

Major problem with the Access Control Matrix Model

- Confinement problem: How to determine whether there is any mechanism by which a subject authorized to access an object may leak information contained in that object to some other subjects not authorized to access that object.
- Another disadvantage is that no semantics of information in the objects are considered; thus the security sensitivity of an object is hardly expressed by that model.

Multilevel Security

- Hierarchy: Top Secret, Secret, Confidential, …
- Information must n't leak from High down to Low
- Enforcement must be independent of user actions!
- Perpetual problem: careless staff
- 1970s worry: operating system insecurity
- 1990s worry: virus at Low copies itself to High and starts signalling down (e.g. covert channel)

Multilevel Security Policy

- Mandatory security policies enforce access control on the basis of regulations mandated by a central authority
- The most common form of mandatory policy is the *multilevel security policy*, based on the classifications of *subjects* and *objects* in the system
- Objects are passive entities storing information
- Subjects are active entities that request access to the objects

- There is a distinction between *subjects* of the mandatory policy and the *authorization subjects* considered in the discretionary policies
- While authorization subjects typically correspond to users (or groups thereof), mandatory policies make a distinction between *users* and *subjects*
- Users are human beings who can access the system, while subjects are processes (i.e., programs in execution) operating on behalf of users
- This distinction allows the policy to control the indirect accesses (leakages or modifications) caused by the execution of processes

Multi-Level Security (MLS)

- Sensitive information be disclosed only to authorized personnel
- Paper World: assign each document and each employee a security level indicating sensitivity and authority.
- Levels: Unclassified, confidential, secret, top_secret
- Levels form a partially ordered set: an employee is authorized to read a document only if his level is greater than or equal to that of the document

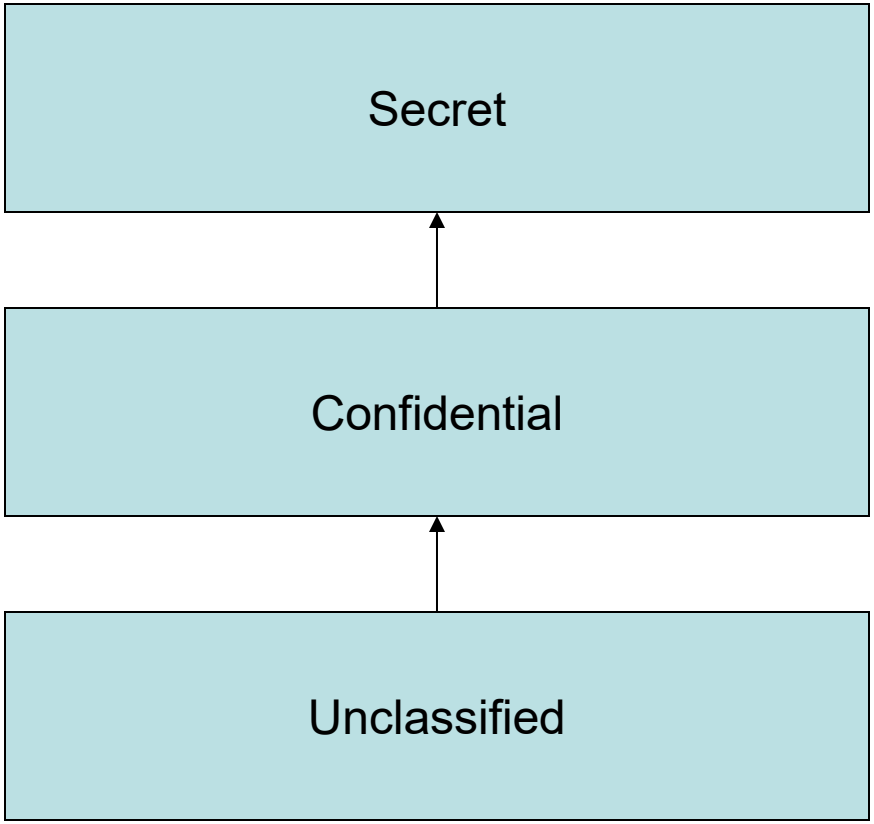
MLS for Information Systems

- Not all information is in the form of documents
- Not all consumers are employees
- Two Aspects:
  - Access Control: Determining who can see information of a given sensitivity leaving the system.
  - Correct Labeling: Determining the sensitivity of information entering and leaving the system.
    - Important in the context of Trojan Horses
- Challenge: Build systems that are secure even in the presence of malicious programs

Security Classifications

- In multilevel mandatory policies, an access class is assigned to each object and subject
- The access class is one element of a partially ordered set of classes
- The partial order is defined by a *dominance* relationship, which we denote with ≥
- In the most general case, the set of access classes can simply be any set of labels that together with the dominance relationship defined on them form a POSET (partially ordered set)

Information Flows

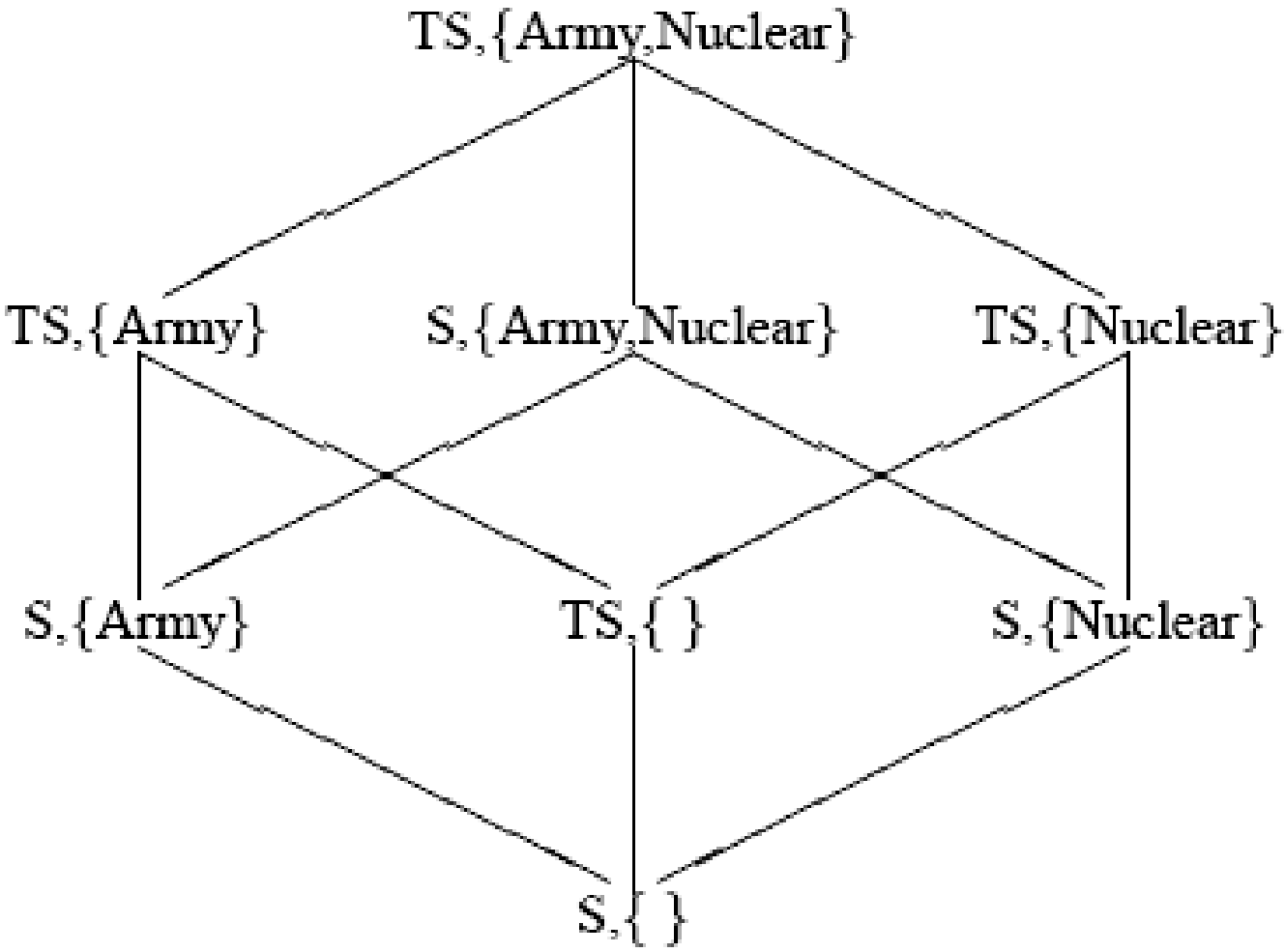


- Most commonly an access class is defined as consisting of two components: a *security level* and a *set of categories*
- The security level is an element of a hierarchically ordered set
- The set of categories is a subset of an unordered set, whose elements reflect functional, or competence areas
- The dominance relationship ≥ is then defined as follows: an access class c_1 dominates (≥) an access class c_2 iff the security level of c_1 is greater than or equal to that of c_2 and the categories of c_1 include those of c_2

- Formally, given a totally ordered set of security levels L, and a set of categories C, the set of access classes is $AC=L \times 2^C$, and $\forall c_1 = (L_1,C_1)$, $c_2 = (L_2,C_2)$: $c_1 \geq c_2 \Leftrightarrow L_1 \geq L_2 \wedge C_1 \supseteq C_2$
- Two classes c_1 and c_2 such that neither $c_1 \geq c_2$ nor $c_2 \geq c_1$ holds are said to be incomparable
- It is easy to see that the dominance relationship so defined on a set of access classes AC satisfies the following properties
  - reflexivity, transitivity, antisymmetry
  - Existence of a least upper bound (LUB) and a greatest lower bound (GLB)

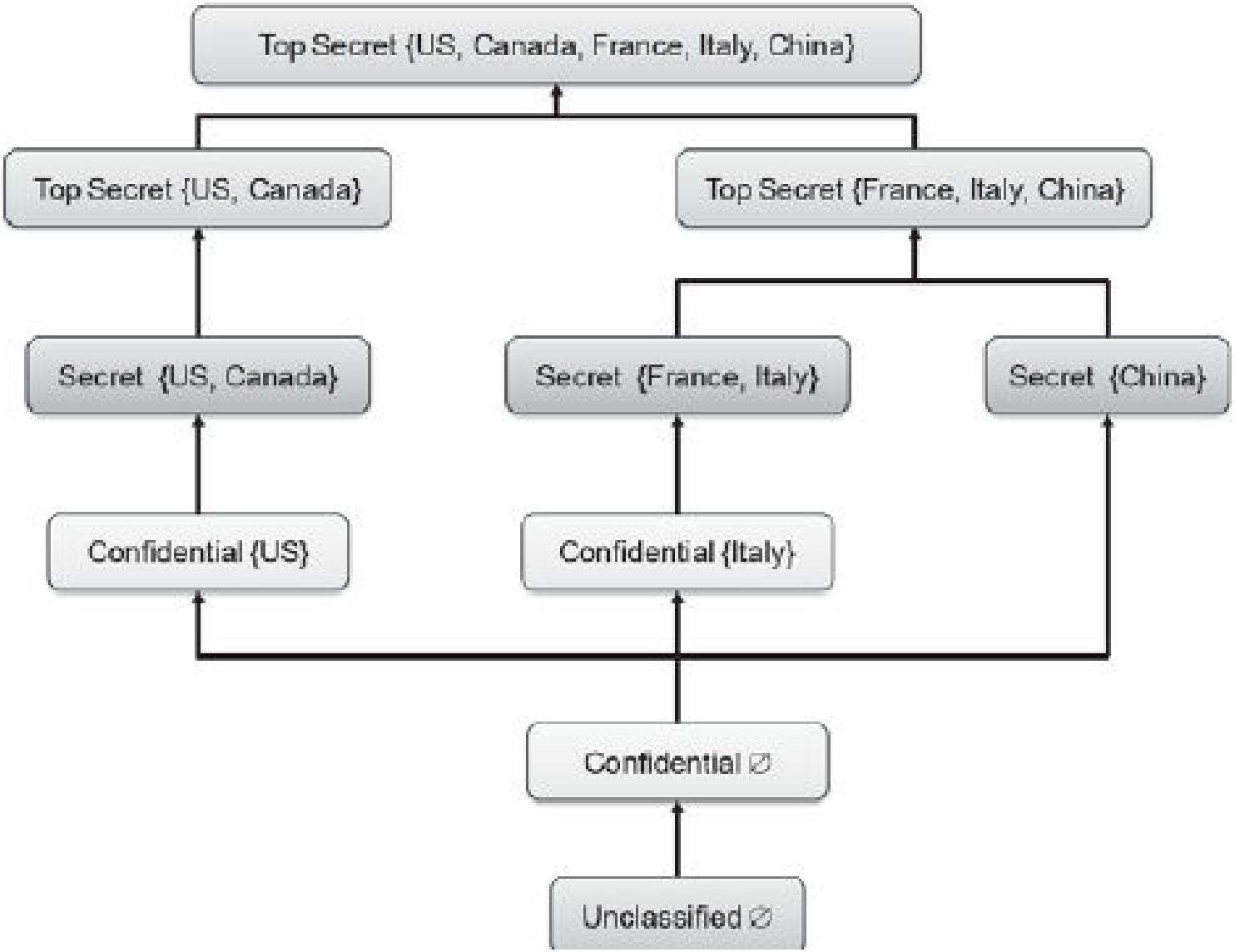
- Access classes defined as above together with the dominance relationship between them therefore form a *lattice*

- The semantics and use of the classifications assigned to objects and subjects within the application of a multilevel mandatory policy is different depending on whether the classification is intended for a *secrecy* or an *integrity* policy



An example security lattice

Defining Security Levels using Categories



Secrecy-based mandatory policies

- A secrecy mandatory policy controls the *direct* and *indirect* flows of information to the purpose of preventing leakages to unauthorized subjects
- The security level of the access class associated with an object reflects the sensitivity of the information contained in the object, that is, the potential damage that could result from the unauthorized disclosure of the information
- The security level of the access class associated with a user, also called *clearance*, reflects the user's trustworthiness not to disclose sensitive information to users not cleared to see it

- Categories define the area of competence of users and data and are used to provide finer grained security classifications of subjects and objects than classifications provided by security levels alone. They are the basis for enforcing *need-to-know* restrictions
- Users can connect to the system at any access class dominated by their clearance
- A user connecting to the system at a given access class originates a subject at that access class

The Lattice Model

- The best-known Information Flow Model
- Based upon the concept of lattice whose mathematical meaning is a structure consisting of a finite partially ordered set together with a least upper bound and greatest lower bound operator on the set.
- Lattice is a Directed Acyclic Graph(DAG) with a single source and sink.
- Information is permitted to flow from a lower class to upper class.

The lattice model (continued)

A lattice model of secure information flow is the 7-tuple $\langle S, O, SC, F, \oplus, \otimes, \rightarrow \rangle$

- S set of subjects
- O set of objects
- SC poset of security classes
- F the *binding* function $F : S \cup O \rightarrow SC$
- $\oplus$ The infimum operator on SC
- $\otimes$ The supremum operator on SC
- $\rightarrow$ The flow relation on pairs of security classes.

The lattice model (continued)

- This satisfies the definition of lattice. There is a single source and sink.
- The least upper bound of the security classes {x} and {z} is {x,z} and the greatest lower bound of the security classes {x,y} and {y,z} is {y}.

Flow Properties of a Lattice

- The relation → is reflexive, transitive and antisymmetric for all A,B,C **ε** SC.
- Reflexive: A → A
  - Information flow from an object to another object at the same class does not violate security.
- Transitive: A → B and B → C implies A → C .
  - This indicates that a valid flow does not necessarily occur between two classes adjacent to each other in the partial ordering
- Antisymmetric: A → B and B → A implies A=B
  - If information can flow back and forth between two objects, they must have the same classes

Flow Properties of a Lattice (Contd..)

- Two other inherent properties are as follows

- Aggregation: A → C and B → C implies A U B → C
  - If information can flow from both A and B to C , the information aggregate of A and B can flow to C.

- Separation: A U B → C implies A → C and B → C
  - If the information aggregate of A and B can flow to C ,information can flow from either A or B to C

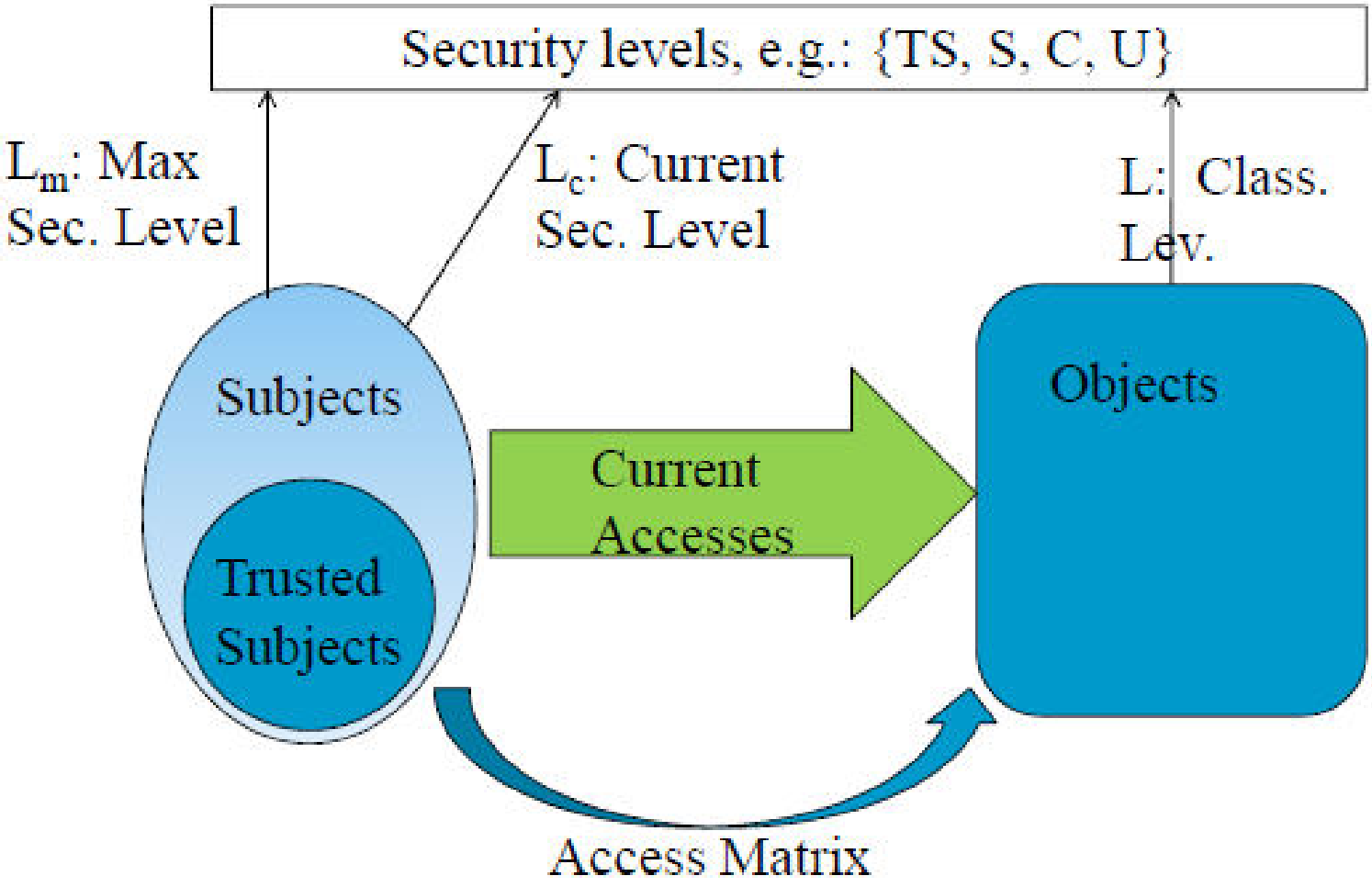
What is a Security Model?

- A model describes the system
  - e.g., a high level specification or an abstract machine description of what the system does
- A security policy
  - defines the security requirements for a given system
- Verification shows that a policy is satisfied by a system
- System Model + Security Policy = Security Model

Multilevel Security Models

- Multilevel Security is a special case of the lattice-based information flow model. There are two well-known multilevel security models:
- The Bell-LaPadula Model Focuses on confidentiality of information
- The Biba Model Focuses on system integrity

Elements of the BLP Model



The Bell-LaPadula Model

- L is a linearly ordered set of security levels
- C is a lattice of security categories

- The security class assigned to a subject or an object includes two components: a hierarchical security level and a nonhierarchical security category.

- The security level is called the clearance if applied to subjects, and classification if applied to objects.

- Each security category is a set of compartments that represent natural or artificial characteristics of subjects and objects and is used to enforce the need-to-know principle.

The Bell-LaPadula Model contd…

- Need-to-know principle: A subject is given access only to the objects that it requires to perform its jobs.

- The lattice of security classes is L × C. If AB **ε** SC, A dominates B if A's level is higher than B's level and B's category is a subset of A's category.

The Bell-LaPadula Model contd…

- Security with respect to confidentiality in the Bell-LaPadula model is described by the following two axioms:
- **Simple security property**: Reading information from an object o by a subject s requires that SC(s) dominates SC(o) ”no read up”).

- **The *-property**: Writing information to an object o by a subject s requires that SC(o) dominates SC(s).

- Note: In * property , information cannot be compromised by exercising a Trojan Horse program(A code segment that misuses its environment is called a Trojan Horse).
- Example of Trojan Horse: Email attachments

The need-to-know principle

- Even if someone has all the necessary official approvals (such as a security clearance) to access certain information they should not be given access to such information unless they have a *need to know*: that is, unless access to the specific information necessary for the conduct of one's official duties.
- Can be implemented using categories and or DAC

Summarizing BLP

Security Goal of BLP

- There are security classifications or security levels
  - Users/principals/subjects have security clearances
  - Objects have security classifications
- Example
  - Top Secret
  - Secret
  - Confidential
  - Unclassified
- In this case Top Secret > Secret > Confidential > Unclassified
- Security goal (confidentiality): ensures that information do not flow to those not cleared for that level

Approach of BLP

- Use state-transition systems to describe computer systems
- Define a system as secure iff. every reachable state satisfies 3 properties
  - simple-security property, *-property, discretionary-security property
- Prove a Basic Security Theorem (BST)
  - so that one can prove a system is secure by proving things about the system description

The BLP Security Model

- A computer system is modeled as a state-transition system
  - There is a set of subjects; some are designated as trusted.
  - Each state has objects, an access matrix, and the current access information.
  - There are state transition rules describing how a system can go from one state to another
  - Each subject s has a maximal sec level $L_m(s)$, and a current sec level $L_c(s)$
  - Each object has a classification level

The BLP Security Model

- A state is secure if it satisfies
  - Simple Security Condition (no read up):
    - S can read O iff $L_m(S) \geq L(O)$
  - The Star Property (no write down): for any S that is not trusted
    - S can read O iff $L_c(S) \geq L(O)$
    - S can write O iff $L_c(S) \leq L(O)$
  - Discretionary-security property
    - every access is allowed by the access matrix
- A system is secure if and only if every reachable state is secure.

STAR-PROPERTY

- Applies to subjects (principals) not to users
- Users are trusted (must be trusted) not to disclose secret information outside of the computer system
- Subjects are not trusted because they may have Trojan Horses embedded in the code they execute
- Star-property prevents overt leakage of information and does not address the covert channel problem

Is BLP Notion of Security Good?

- The objective of BLP security is to ensure
  - a subject cleared at a low level should never read information classified high
- The ss-property and the *-property are sufficient to stop such information flow at any given state.
- What about information flow across states?

BLP Security Is Not Sufficient!

- Consider a system with s_1, s_2, o_1, o_2
  - $f_S(s_1)=f_C(s_1)=f_O(o_1)=$high
  - $f_S(s_2)=f_C(s_2)=f_O(o_2)=$low
- And the following execution
  - s_1 gets access to o_1, read something, release access, then change current level to low, get write access to o_2, write to o_2
- Every state is secure, yet illegal information exists

SOLUTION: TRANQUILITY PRINCIPLE

•subject cannot change current levels

Objections to BLP (1)

- Some processes, such as memory management, need to read and write at all levels
- Fix: put them in the trusted computing base
- Consequence: once you put in all the stuff a real system needs (backup, recovery, comms, …) the TCB is no longer small enough to be easily verifiable

Objections to BLP(2)

- John MacLean's "System Z": as BLP but lets users req. temporary declassification of any file
- Fix: add tranquility principles
  - Strong tranquility: labels never change
  - Weak tranquility: they don't change in such a way as to break the security policy
- Usual choice: weak tranquility using the "high watermark principle" – a process acquires the highest label of any resource it's touched
- Problem: have to rewrite apps (e.g. license server)

Objections to BLP (3)

- High can't acknowledge receipt from Low
- This blind write-up is often inconvenient: information vanishes into a black hole
- Option 1: accept this and engineer for it (Morris theory) – CIA usenet feed
- Option 2: allow acks, but be aware that they might be used by High to signal to Low
- Use some combination of software trust and covert channel elimination

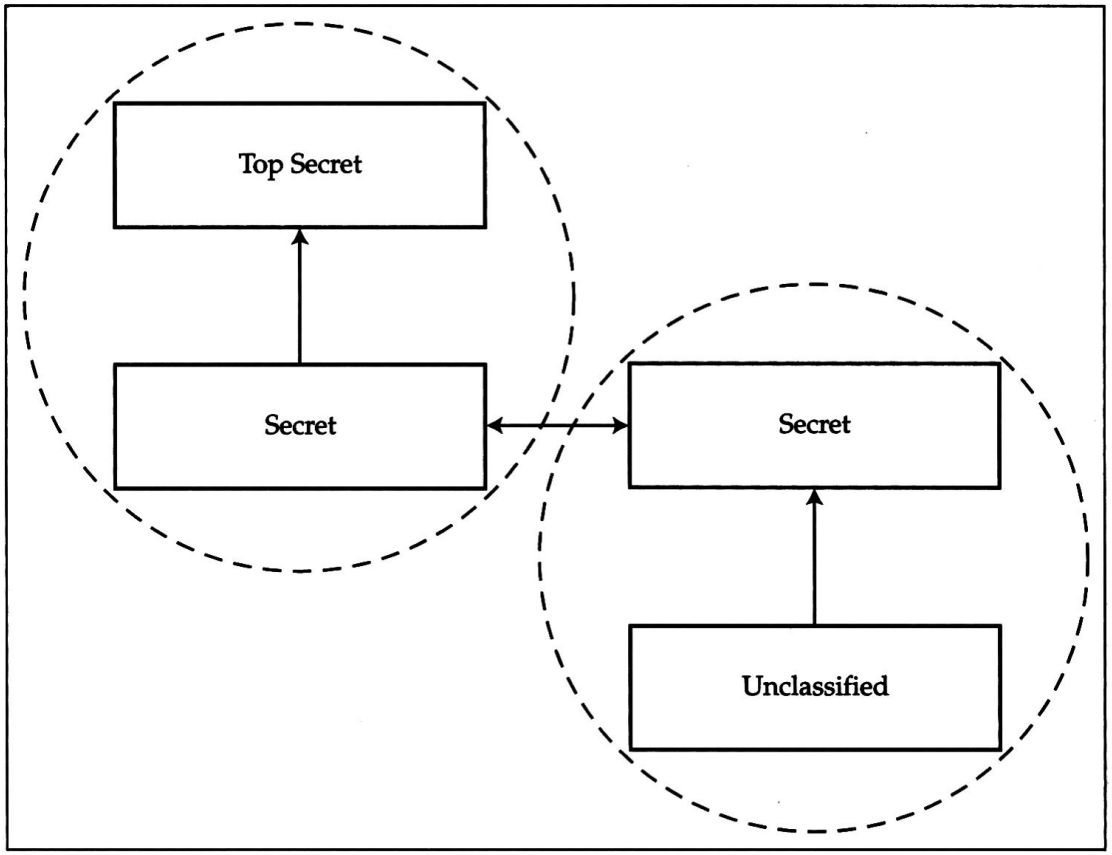
Variants of BLP

- Noninterference: no input by High can affect what Low can see. So whatever trace there is for High input X, there's a trace with High input Ø that looks the same to Low (Goguen & Messeguer 1982)
- Nondeducibility: weakens this so that Low is allowed to see High data, just not to understand it – e.g. a LAN where Low can see encrypted High packets going past (Sutherland 1986)

Variants on Bell-LaPadula (2)

- Biba integrity model: deals with integrity rather than confidentiality. It's "BLP upside down" – high integrity data mustn't be contaminated with lower integrity stuff
- Domain and Type Enforcement (DTE): subjects are in domains, objects have types
- Role-Based Access Control (RBAC): current fashionable policy framework

The Cascade Problem



Ross Anderson

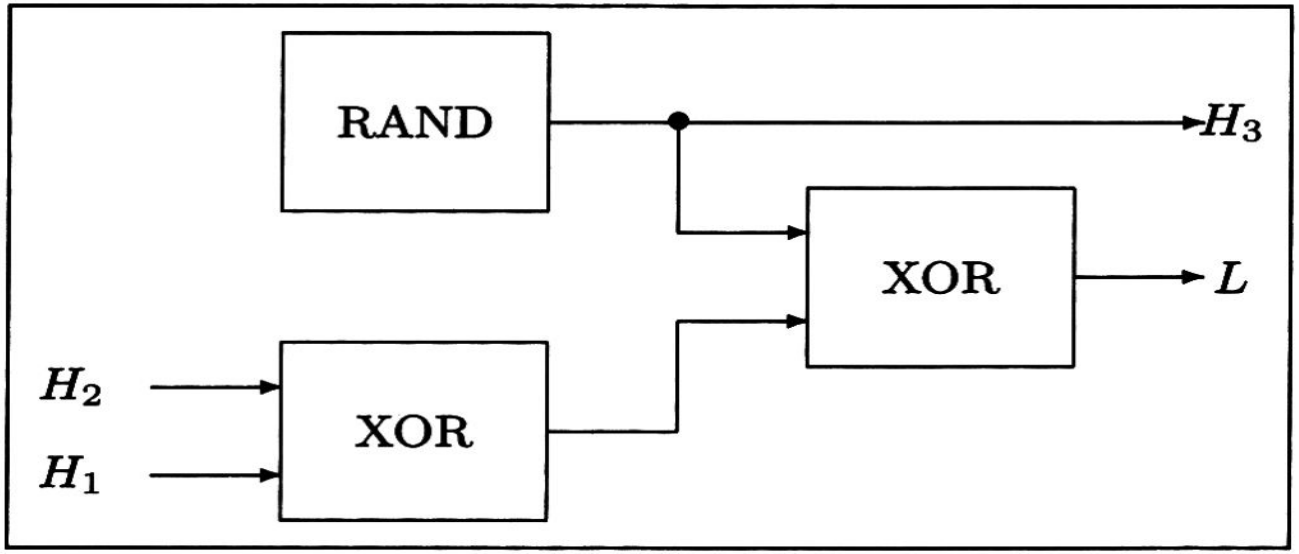
Connecting the two -- issue

- Consider a simple device that accepts two High inputs H1 and H2, multiplexes them, encrypts them by xor'ing them with a one-time pad, outputs the other copy of the pad on H3, and outputs the ciphertext, which being encrypted with a cipher system giving perfect secrecy, is considered to be Low (output L)

- In isolation, this device is provably secure. But if feedback is permitted, then the output from H3 can be fed back into H2, with the result that the high input H1 now appears at the low output L.

Composability

- Systems can become insecure when interconnected, or when feedback is added



Ross Anderson

Composability

- So nondeducibility doesn't compose
- Neither does noninterference
- Many things can go wrong – clash of timing mechanisms, interaction of ciphers, interaction of protocols
- Practical problem: lack of good security interface definitions (Keep in mind API failures)
- Labels can depend on data volume, or even be non-monotone (e.g. Secret laser gyro in a Restricted inertial navigation set)

Consistency

- US approach (polyinstantiation):

	Cargo	Destination
Secret	Missiles	Iran
Unclassified	Spares	Cyprus

- UK approach (don't tell low users):

	Cargo	Destination
Secret	Missiles	Iran
Restricted	Classified	Classified

Downgrading

- A related problem to the covert channel is how to downgrade information
- Analysts routinely produce Secret briefings based on Top Secret intelligence, by manual paraphrasing
- Also, some objects are downgraded as a matter of deliberate policy – an act by a trusted subject
- For example, a Top Secret satellite image is to be declassified and released to the press

Examples of MLS Systems

- SCOMP – Honeywell variant of Multics, launched 1983. Four protection rings, minimal kernel, formally verified hardware and software. Became the XTS-300
- Used in military mail guards
- Motivated the ‘Orange Book’ – the Trusted Computer System Evaluation Criteria
- First system rated A1 under Orange Book

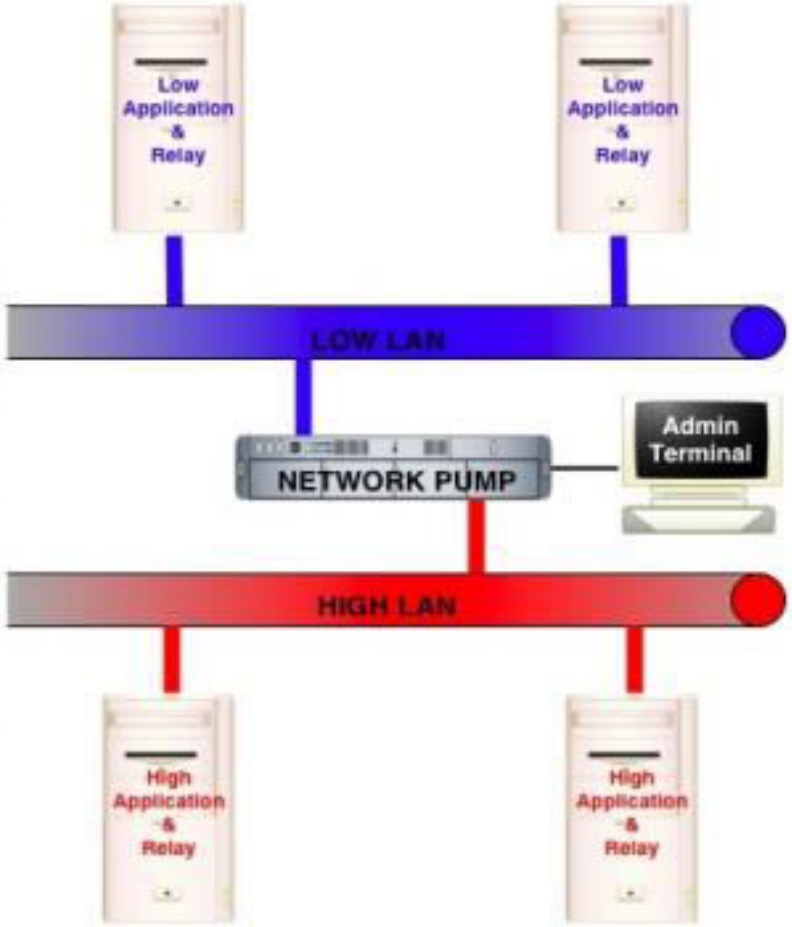
Examples of MLS Systems (2)

- Blacker – series of encryption devices designed to prevent leakage from "red" to "black". Very hard to accommodate administrative traffic in MLS!
- Compartmented Mode Workstations (CMWs) – used by analysts who read Top Secret intelligence material and produce briefings at Secret or below for troops, politicians … Mechanisms allow cut-and-paste from L $\rightarrow$ H, L $\rightarrow$ L and H $\rightarrow$ H but not H $\rightarrow$ L

Examples of MLS Systems (3)

Overview

The Network Pump® is a Government off-the-shelf (GOTS) High Assurance "One-Way" Guard that enables applications operating on a lower security level network to pass information to applications on a highe security level network automatically Developed by the U.S. Naval Research Laboratory (NRL), Center for High Assurance Computer Systems, the "One-Way" Guard delivers information without leakage from the High network to the Low network.



Examples of MLS Systems (4)

- LITS – RAF Logistics IT System – a project to control stores at 80 bases in 12 countries. Most stores ‘Restricted’, rest ‘Secret’, so two databases connected by a pump
- Other application-level rules, e.g. ‘don’t put explosives and detonators in the same truck’
- Eventual solution: almost all stuff at one level, handle nukes differently

Examples of MLS Systems (5)

- DERA's 'Purple Penelope' was an attempt to relax MLS to accountability for lower levels of stuff
- Driver: people determined to use Office
- Solution: wrapper around Windows that tracks current level using high watermark
- Downgrading allowed, but system forces authentication and audit
- Now called 'Sybard Suite'