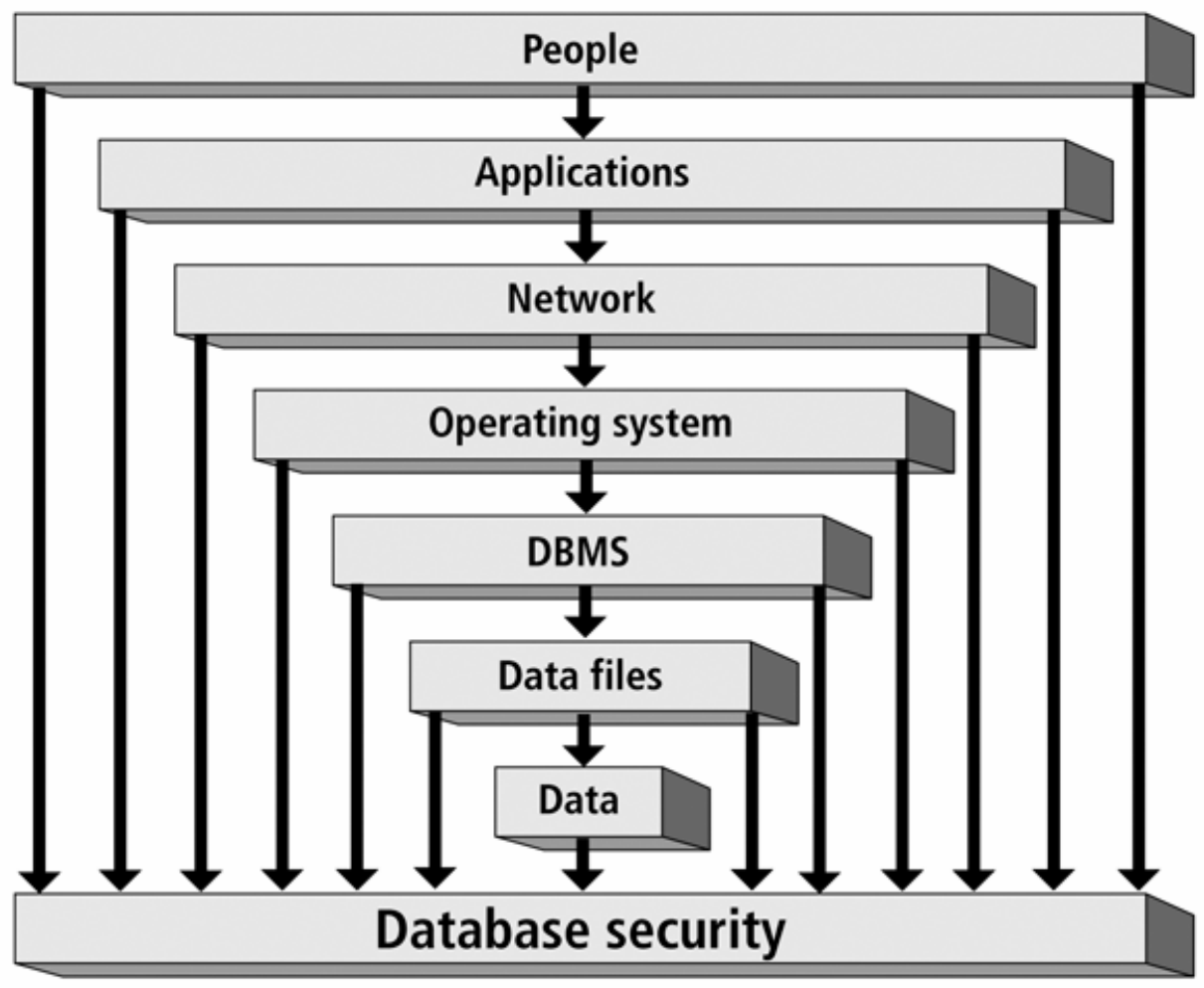
Administration of Users

(055049)

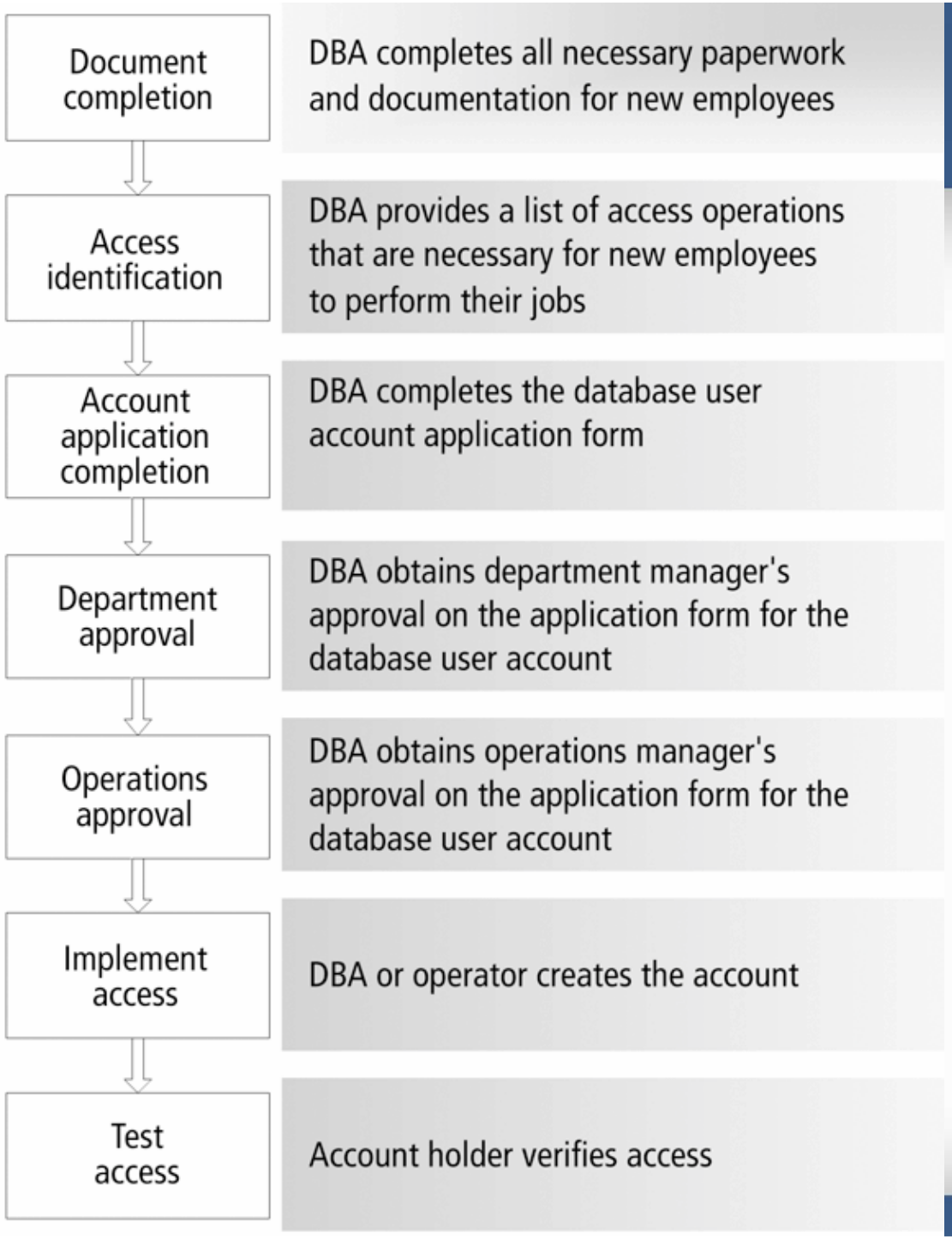
Truong Tuan Anh
CSE-HCMUT

Context


People
Applications
Network
Operating system
DBMS
Data files
Data
Database security

User Administration Documentation

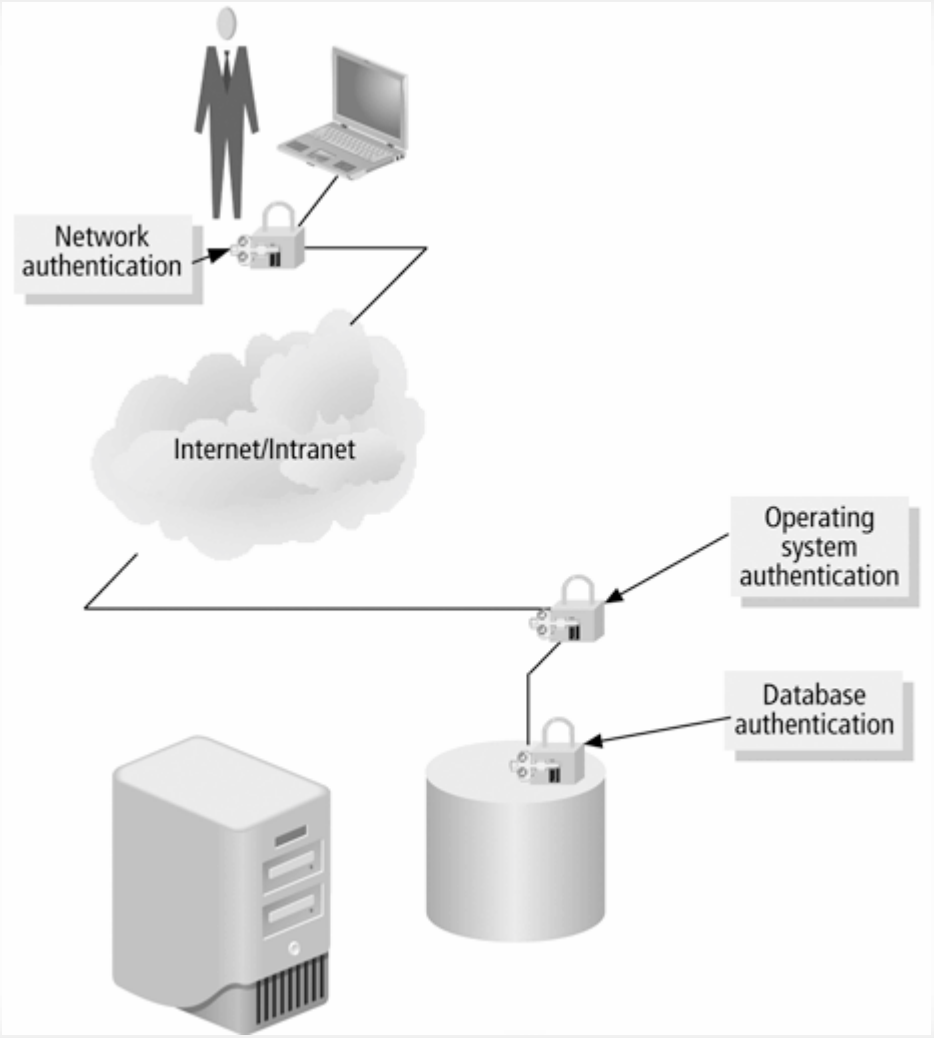
- Part of the administration process
- Reasons to document:
  - Provide a paper trail
  - Ensure administration consistency
- What to document:
  - Administration policies, staff and management
  - Security procedures
  - Scripts or programs
  - Predefined roles description


Document completion
DBA completes all necessary paperwork and documentation for new employees
Access identification
DBA provides a list of access operations that are necessary for new employees to perform their jobs
Account application completion
DBA completes the database user account application form
Department approval
DBA obtains department manager's approval on the application form for the database user account
Operations approval
DBA obtains operations manager's approval on the application form for the database user account
Implement access
DBA or operator creates the account
Test access
Account holder verifies access

Operating System Authentication

- Many databases depend on OS to authenticate users
- Once an intruder is inside the OS, it is easier to access the database
- Centralize administration of users
- Ideally, users must be authenticated at each level

Multi-level Authentication

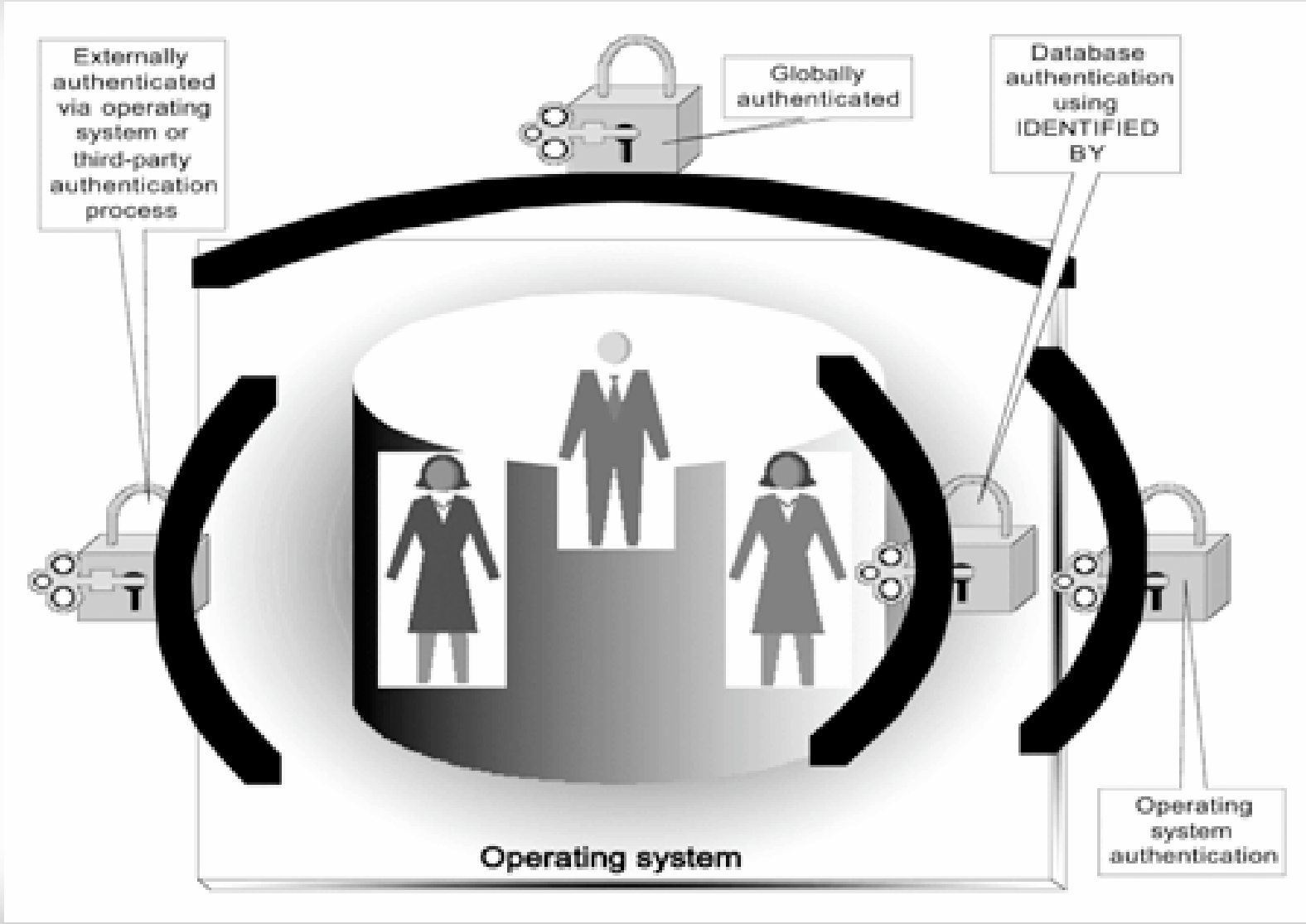


User Administration in Databases

Creating Users

- Must be a standardized, well-documented, and securely managed process
- Example
  - Several ways in Oracle:
    1. CREATE USER Statement from iSQLPlus
    2. Oracle Enterprise Manager: GUI administration tool using database authentication
    3. Creating an Oracle User Using External (Operating System) Authentication
    4. SQL developer

Creating Users: Oracle



Removing Users

- Simple process
- Make a backup first
- Obtain a written request (for auditing purposes)
- Example: Oracle
  - DROP command
  - CASCADE option: when user owns database objects

```
DROP USER MELVIN CASCADE;
```

Modifying Users

- Modifications involve:
  - Changing passwords
  - Locking an account
  - Increasing a storage quota
- Example: Oracle
  - ALTER USER statement
  - Oracle Enterprise Manager: graphical tool

Default Users

- Oracle default users:
  - SYS, owner of the data dictionary
  - SYSTEM, default DBA, can perform almost all database tasks
- SQL Server default users:
  - SA, system administrator
  - BUILT_IN\Administrators

Best Practices

- Follow company's policies and procedures
- Always document and create logs
- Educate users
- Keep abreast of database and security technology
- Review and modify procedures

- Block direct access to database tables
- Limit and restrict access to the server
- Use strong passwords
- Patches, patches, patches

Creating, Assigning, and Revoking User Roles

- For authorization
- Role:
  - Used to organize and administer privileges
  - It is like a user, except it cannot own object
  - Can be assigned privileges
  - Can be assigned to users

Creating, Assigning, and Revoking User Roles

- Example: in Oracle
  - Create a role using CREATE ROLE statement
  - Assign a role using GRANT statement
  - Revoke a role using REVOKE statement
  - Drop a role using DROP statement

```
CREATE ROLE DEV_ROLE;
GRANT CREATE SESSION TO DEV_ROLE
GRANT DEV_ROLE TO ALICE
```

Multilevel Security (MLS)

Multilevel Security

- Multilevel security (MLS) involves a database in which the data stored has an associated classification and consequently constraints for their access
- MLS allows users with different classification levels to get different views from the same data
- MLS cannot allow **downward leaking**, meaning that a user with a lower classification cannot view data stored with a higher classification

Multilevel Security

- In relational model, relations are tables and relations consist of tuples (rows) and attributes (columns)
- Example:

  Consider the relation

  SOD(Starship, Objective, Destination)

Starship	Objective	Destination
Enterprise Voyager	Exploration Spying	Talos Mars

Multilevel Security

- The relation in the example has *no classification* associated with it in a relational model
- An example in MLS with classification will be as follows:

Starship		Objective		Destination	
Enterprise	**U**	Exploration	**U**	Talos	**U**
Voyager	**U**	Spying	**S**	Mars	**S**

Multilevel Security

- In MLS, access classes can be assigned to:
  - Individual tuple in a relation
  - Individual attribute of a relation
  - Individual data element of tuples in a relation
- Bell – LaPadula Model
  - Secrecy-Based Mandatory Policies
- Biba Model
  - Integrity-based Mandatory Policies

Bell – LaPadula Model (BLP)

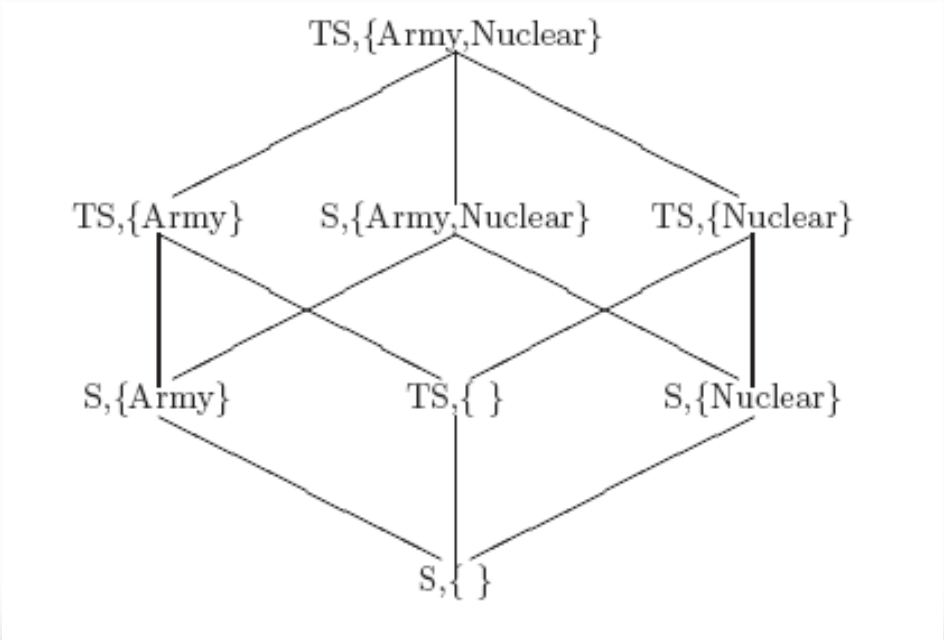
- Bell-LaPadula model was developed in 1973
- This is an extension of the Access Matrix model with classified data
- This model has two components:
  - Classification
  - Set of categories

Bell – LaPadula Model (BLP)

- Classification has four values {U, C, S, TS}
  - U = unclassified
  - C = confidential
  - S = secret
  - TS = top secret
- Classifications are ordered: TS > S > C > U
- Set of categories consists of the data environment and the application area, i.e., Nuclear, Army, Financial, Research
- Example: In USA, a “SECRET” clearance involves checking FBI fingerprint files

Bell – LaPadula Model (BLP)

- An access class c1 **dominates** ≥ an access class c2 iff
  - Security level of c1 is **greater** than or equal to that of c2
  - The categories of c1 **include** those of c2



Bell – LaPadula Model (BLP)

- Bell-LaPadula model is based on a subject-object paradigm
- Subjects are active elements of the system that execute actions
- Objects are passive elements of the system that contain information
- Subjects act on behalf of users who have a security level associated with them (indicating the level of system trust)
- Subjects and objects are assigned access classes

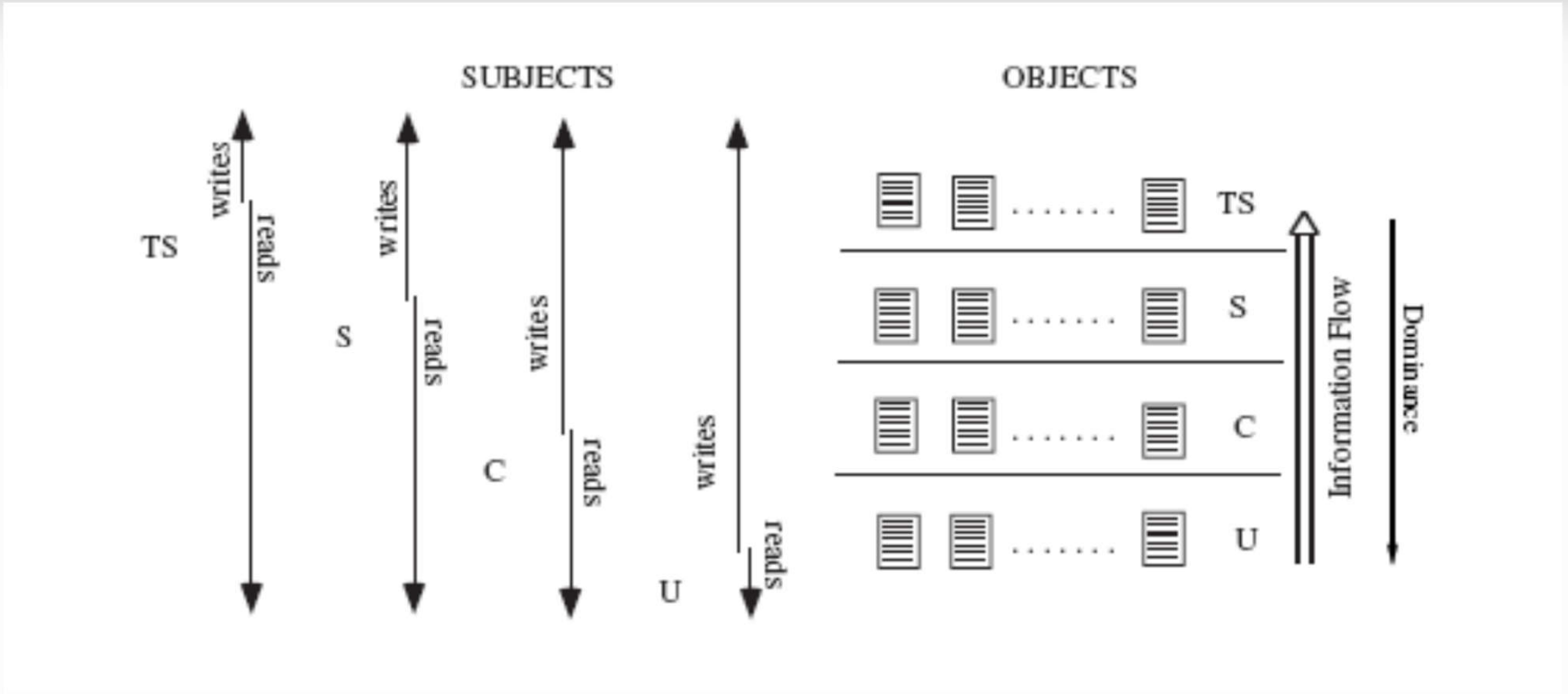
Bell – LaPadula Model (BLP)

- Subjects execute access modes on objects
- Access modes are:
  - Read-only
  - Append (writing without reading)
  - Execute
  - Read-write (writing known data)

Bell – LaPadula Model (BLP)

- To protect information **confidentiality**
  - **No-read-up**, a subject is allowed a read access to an object only if the access class of the subject dominate the access class of the object
  - **No-write-down**, a subject is allowed a write access to an object only if the access class of the subject is dominated by the access class of the object

No-read-up & No-write-down



- Can TS subject write to S object?
- Can S subject write to U object?
- How to apply to the Trojan Horse case?

Bell – LaPadula Model (BLP)

- Two main properties of this model for a secure system are:
  - Simple security property
  - Star property
- **Simple security** means: a subject at a given security level may not read an object at a higher security level (*no read-up*)
- **Star property** means: a subject at a given security level must not write to any object at a lower security level (*no write-down*)

BLP: Problem

- If I can write up, then how about writing files with blanks?
  - Blind writing up may cause integrity problems, but not a confidentiality breach

Bell – LaPadula Model

- This model guarantees secrecy by preventing unauthorized release of information
- This model does not protect from unauthorized modification of information

The Biba Model

- A model due to Ken Biba which is often referred to as "*Bell-LaPadula upside down*"
- It deals with integrity alone and ignores confidentiality entirely
- Each subject and object in the system is assigned an integrity classification
  - Crucial
  - Important
  - Unknown

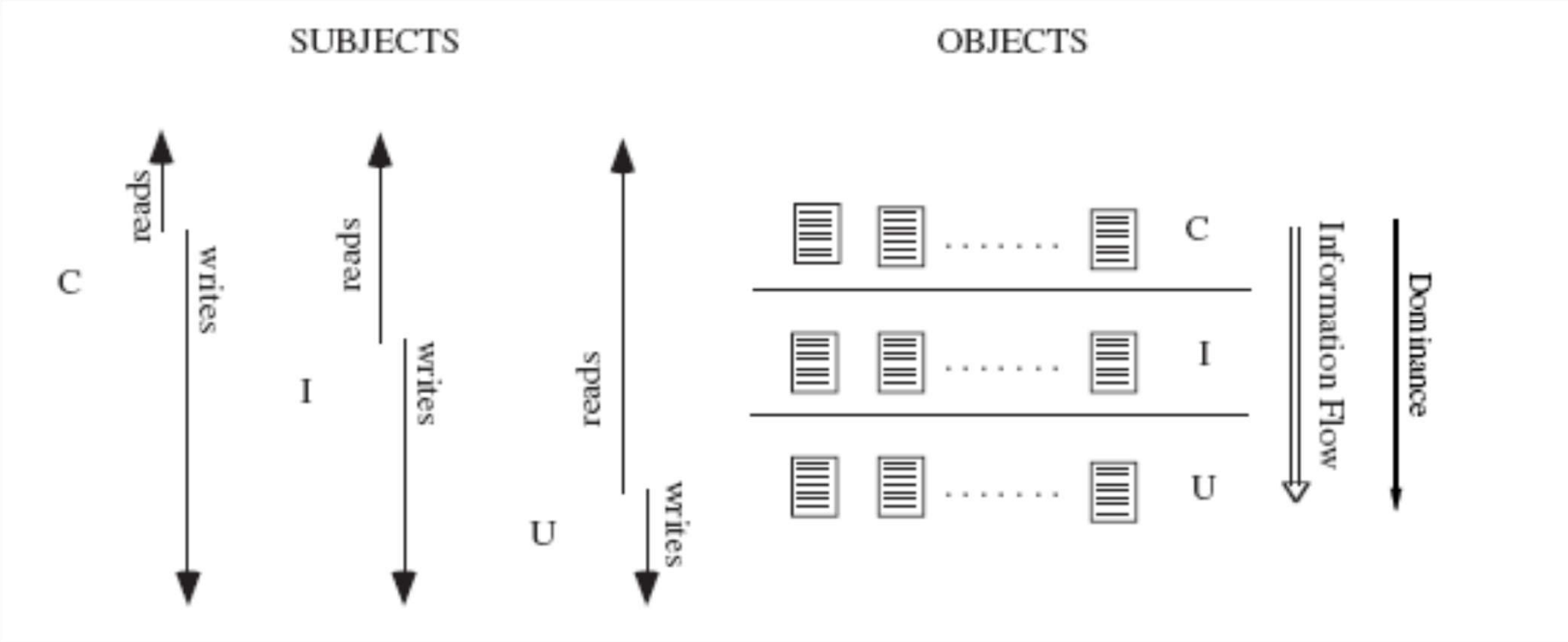
Integrity Level

- **Integrity level** of a user reflects user's trustworthiness for *inserting*, *modifying*, or *deleting* information
- **Integrity level** of an object reflects both the degree of trust that can be placed on the info stored in the object, and the potential damage could result from unauthorized modification of info

Two Principles

- **No-read-down**: A subject is allowed a *read* access to an object only if the integrity level of the object dominates the integrity level of the subject
- **No-write-up**: A subject is allowed a *write* access to an object only if the integrity level of the object is dominated by the integrity level of the subject

Two Principles



Q: How to control both the secrecy and integrity?

Applying to Databases

- Commercial DBMSs Oracle, Sybase, and TruData have MLS versions of their DBMS
- Because of Bell-LaPadula restrictions, subjects having different clearances see different versions of a multilevel relation

Name	λ_N	Dept	λ_D	Salary	λ_S
Bob	U	Dept1	U	100K	U
Jim	U	Dept1	U	100K	U
Ann	S	Dept2	S	200K	S
Sam	U	Dept1	U	150K	S

(a)

Visible to a user with secret level.

Name	λ_N	Dept	λ_D	Salary	λ_S
Bob	U	Dept1	U	100K	U
Jim	U	Dept1	U	100K	U
Sam	U	Dept1	U	–	U

(b)

Visible to a user with unclassified level.

PolyInstantiation

- Request by low level subject
  - An **u**nclassified subject requests insert of <Ann, Dept1, 100K>
- If this update is rejected, then the user would be able to infer something about Ann
- MLS would allow the secret channel to permit data update and protect data integrity

Name	λ_N	Dept	λ_D	Salary	λ_S
Bob	U	Dept1	U	100K	U
Jim	U	Dept1	U	100K	U
Ann	S	Dept2	S	200K	S
Sam	U	Dept1	U	150K	S
Ann	U	Dept1	U	100K	U
Sam	U	Dept1	U	100K	U

(a)

Visible to a user with secret level.

Name	λ_N	Dept	λ_D	Salary	λ_S
Bob	U	Dept1	U	100K	U
Jim	U	Dept1	U	100K	U
Ann	U	Dept1	U	100K	U
Sam	U	Dept1	U	100K	U

(b)

Visible to a user with unclassified level.

Challenges

- Cover Stories
  - Non-true data to hide the existence of the actual value
  - Not released is a cause of information leakage
- Fine-grained is not easy
  - Aggregation, association
  - Block inference channels

Multilevel DBMSs Architecture

- Trusted subject. The DBMS itself must be trusted to ensure mandatory policy
- Trusted Computing Base: Data are partitioned in different databases, one for each level