The Jajodia & Sandhu model

- **Jajodia & Sandhu (1991)**, a model for the application of mandatory policies in relational database systems. Based on the sec classifications introduced in BLP. It extends the standard relational model to consider the sec classification.

- **Multilevel relations**: Schema and multiple instances based on each access class. A multilevel relation consists of two parts:

1. A state-independent multilevel relation scheme $R(A_1, C_1, \ldots, C_n, TC)$, where each $A_i$ is a data attribute defined over domain $D_i$, each $C_i$ is a classification attribute for $A_i$, and $TC$ is the tuple class attribute.

   - The domain of $C_i$ is specified by a range $[L_i, H_i]$ which is specified as a sub-lattice of access classes.
   - The domain of $TC$ is $[\text{lub}(L_i), \text{lub}(H_i)]$.

2. A collection of state-dependent relation instances $R_c(A_1, C_1, \ldots, A_n, C_n, TC)$, one for each access class $c$ in the given lattice; each instance is a set of distinct tuples of the form $(a_1, c_1, \ldots, a_n, c_n, tc)$ where each element $a_i$ is either a value of domain $D_i$ or null, each $c_i$ is a value of the specified range and smaller than $tc$, that is, $c_i \in [L_i, H_i]$ $c_i\text{tc}$, and $tc$ is the least upper bound of the classes of the attribute in the tuple: that is, $tc = \text{lub}\{c_i : i=1, \ldots, n\}$

**Example of a multilevel relation Employee**

<table>
<thead>
<tr>
<th>Name</th>
<th>$C_{Name}$</th>
<th>Department</th>
<th>Salary</th>
<th>$C_{Salary}$</th>
<th>TC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bob</td>
<td>S</td>
<td>Dept1</td>
<td>$1K$</td>
<td>$S$</td>
<td>$S$</td>
</tr>
<tr>
<td>Ann</td>
<td>S</td>
<td>Dept2</td>
<td>$2K$</td>
<td>$TS$</td>
<td>$TS$</td>
</tr>
<tr>
<td>Sam</td>
<td>TS</td>
<td>Dept2</td>
<td>$3K$</td>
<td>$TS$</td>
<td>$TS$</td>
</tr>
</tbody>
</table>

**Properties of the model:**

1. **Entity integrity**: Let $AK$ be the apparent key of a relation $R$. A multilevel relation $R$ satisfies entity integrity if, and only if, for all instances $R_c$ of $R$ and $1 < R_c$ $1 < R_c$

   - (1) $A_i \in AK \Rightarrow |A_i| \text{null}$
   - (2) $A_i, A_j \in AK \Rightarrow |C_i| = |C_j|$, i.e., $AK$ is uniformly classified, and
   - (3) $A_i \in AK \Rightarrow |C_{AK}| = |C_{AK}|$ (where $C_{AK}$ is defined as the classification of the apparent key)
Null values!

- Null values have two meanings:
  - Corresponding to real null values or
  - To attributes at a classification higher than the classification of the instance.

- Two similar value tuples with different attribute security class (so hidden, turned to null)!

- Subsumption relationship: t subsumes s, if for every attribute Ai:
  - t[AI, Ci] = s[AI, Ci] or
  - t[AI] = null and s[AI] = null.

The Jajodia & Sandhu model (cont.)

Properties of the model (cont.):

(2) Null integrity: A multilevel relation R satisfies null integrity if and only if for each instance Rc of R both the following conditions are satisfied:

   1. For all t ∈ Rc, t[AI] = null ⇒ t[Cj] = t[CAK]: that is, null values are classified at the level of the key.
   2. Rc is subsumption free in the sense that it does not contain two distinct tuples such that one subsumes the other.

   A tuple t subsumes s if for every attribute Ai
   - t[AI, CI] = s[AI, CI] or
   - t[AI] = null and s[AI] = null.

3) Inter-instance integrity

Controlling the consistency among the different instances of a relation

A multilevel relation R satisfies inter-instance integrity if and only if for all c′ ≤ c, Rc = σ(Rc, c′), where the filter function σ produces the c′-instance Rc from R as follows:

1. For every tuple t ∈ Rc such that t[CAK] ≤ c′, there is a tuple t′ ∈ Rc with t[AK,CAK] = t[AK,CAK] and for AI ≠ AK
   - t′[AI, CI] = t[AI, CI] if t[CI] ≤ c′;
   - <null, CAK>: otherwise

Inter-instance integrity (cont.):

A tuple t subsumes s if for every attribute Ai
- t[AI, CI] = s[AI, CI] or
- t[AI] = null and s[AI] = null.

(4) Polyinstantiation integrity property:

A multilevel relation R satisfies Polyinstantiation integrity iff, for every Rc, for all A;

(AK, CAK, CI) → A. That is, the apparent key, together with the classification of the key and the classification of the attribute functionally determines the value of this attribute.

Informally: null integrity and interinstance integrity ensure that, if a tuple value at some security level can be filtered or derived from a higher-classified tuple, then it is sufficient to store the higher classified tuple in the multi-level relation.

Access to Multilevel relations:
- Deal with the write operations (Insert, Update, Delete)
- Read is processed through the Read-Down principle.
The Jajodia & Sandhu model (cont.)

Insert operation:
The insert operation, from a c-user, has the following form:
```
INSERT INTO R, [A, A, ...]
VALUES (a, a, ...)
```
The insert operation is granted, if and only if, the following conditions are satisfied:
1. \( t[AK] \) does not contain any nulls
2. For all \( u \in R, u[AK] \neq t[AK] \)
If the conditions are satisfied, the tuple is inserted into \( R \) and all the instances \( R_{C \leq C} \).

Results of the operation INSERT VALUES " John, Dept2,20K" on S and TS instances of Employee from S subject

<table>
<thead>
<tr>
<th>Name</th>
<th>Dept</th>
<th>Position</th>
<th>Salary</th>
</tr>
</thead>
<tbody>
<tr>
<td>John</td>
<td>Dept2</td>
<td></td>
<td>20K</td>
</tr>
</tbody>
</table>

The Jajodia & Sandhu model (cont.)

Update operation:
An update operation from a c-user has the following form:
```
UPDATE R
SET Ai = Si, A j = S j...
[WHERE P]
```
Where each \( s_i \) is a scalar expression, and \( p \) is a predicate expression which identifies those tuples in \( R \) that are to be modified.
If the conditions are satisfied, the update is propagated into \( R_{C \geq C} \) according to the minimum propagation delay policy: only those tuples which are needed to preserve the inter-instance property are inserted in \( R_{C > C} \).

Results of the operation UPDATE salary = "30K" WHERE Name = "Ann" on S and TS instances of Employee from TS subject

<table>
<thead>
<tr>
<th>Name</th>
<th>Dept</th>
<th>Position</th>
<th>Salary</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ann</td>
<td>Dept2</td>
<td></td>
<td>30K</td>
</tr>
</tbody>
</table>

Delete

- Propagation of Delete to \( R_{C > C} \) due to
  `DELETE FROM R` [WHERE P]
  - If \( t[AK] = c \), delete any polystanitiated tuple in \( R_{C > C} \)
  - If \( t[AK] < c \), the tuple will continue to exist in all instances \( R_{C \geq C} \).

Results of the operation UPDATE Department= "Dept1" WHERE Name = "Ann" and S and TS instances of Employee from TS subject

<table>
<thead>
<tr>
<th>Name</th>
<th>Dept</th>
<th>Position</th>
<th>Salary</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ann</td>
<td>Dept1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Confinement (تحديد)
- Is the problem of preventing a server from leaking info that the user of the service considers confidential.
- A process that does not store data, cannot leak it.
- Observing the flow of control can deduce info about the input.
- A process than cannot be observed and communicate with the others, cannot leak info.

... Total isolation.

Covert channel
- The problem is that total isolation is impossible. Unconfined processes can transmit data over the shared resources.
- A covert channel is a path of communication that was not designed for communication.
- Transitive confinement. If p is confined to prevent leakage, the calling process q can also be confined to leak.

Covert channel
- Two types of covert channels:
  1. Use of storage to transmit info. the model should control all accesses to the storage. If it fails, covert channel arise.
  2. All processes can obtain a rough idea of time time is a communication channel. All processes can read time and can write time.

Isolation
- Virtual Machine: simulating a computer system. JVM
- Analyzing all actions against leaking of info: SANDBOX
  A sandbox is an environment in which the actions of a process are restricted according to the security policy. E.g. JVM
Views

- Views on top of the base tables.
- A single and effective mechanism to support content-dependent authorization.
- e.g. User B creates table B and wants to grant user A the authorization for just tuples with salary > 1000.
  - What should be done??
  Defining the view “select * from T where a1=1000 and then grant B the read authorization on the view

> Views

- Privileges on views in comparison with privileges on the base tables??
  - Depends on the view semantics.
- Definitely, having a privilege on a view depends on having that on all tables directly referenced by the view. View on a single table or on multiple!
- Depending on the view semantics, the view owner may have more restricted than on the base tables.
- Privileges of the view owner determined at the time of view definition. Timestamp is the time of view definition.
- The grantor of the view, is whom has been assigned as the owner of the view at the definition time.

Implementation of the model

- Two relations named: SYSAUTH and SYSCOLAUTH
- SYSAUTH has the following attributes:
  - UserId: who has the authority
  - Tname
  - Type: R or V
  - Granter
  - Read: The time at which the grantor granted the read privilege. Default is 0
  - Insert: The time …
  - Delete: The time …
  - Update: the columns on which the privilege is granted. It may have All or None or Some values
  - Grantopt: the Grant operation ….

> Implementation of the model

- In the revised version, if two similar grants happened, two records are inserted with different timestamps.
- SYSCOLAUTH: If there is specified “some” in the SYSAUTH update column, a tuple is needed for each column
  - UserId
  - Table
  - Column
  - Granter
  - Grantopt

Extension of the model

- 1982 by Wilms and Linsday to consider group management.
- Set of users in a group.
- Groups may overlap.
- Applied in System R*, the distributed DBMS.
- Another extension: having cascadable revoke or non-cascadable revoke!
Main features of a Secure DB Arch.

- SDBMSs operate according to two possible modes:
  - System-High or Multilevel
- In System-High DBMSs, all the users are cleared to the highest security.
- Before releasing data, a guard must review such data in order to properly release them.
- This model has the risk for security, as all users are cleared to the highest clearance level.
- It is simple!! Can use existing DBMSs.

Multi-level mode

- Based on using trusted and untrusted DBMSs.
  - Trusted Subject Archs
    - Using a trusted DBMS and a trusted OS
    - From scratch or extension of the security features.
  - Woods Hole Archs, where an untrusted DBMS is used with an additional trusted filter
    - Integrity Lock
    - Kernelized
    - Replicated

The following is a table showing the differences between architecture and implementation prototypes:

<table>
<thead>
<tr>
<th>Architecture</th>
<th>Research prototype</th>
<th>Commercial DBMS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integrity Lock</td>
<td>Mitre</td>
<td>TRU DATA</td>
</tr>
<tr>
<td>Kernelized</td>
<td>SeaView</td>
<td>Oracle</td>
</tr>
<tr>
<td>Replicated</td>
<td>NRL</td>
<td></td>
</tr>
<tr>
<td>Trusted Subject</td>
<td>A1 Secure DBMS</td>
<td>Sybase, Informix, Ingres, Oracle, DEC</td>
</tr>
</tbody>
</table>

Trusted Subject Arch

- Figure in the next slide.
- A set of untrusted front ends is used to interface with different clearances (Low and High).
- TDBMS and TOS form a single TCB (Trusted Computing Base)
- DBMS is responsible for multi-level protection of DB objects.
- High level dominates the low-level.
- DBMS subjects and objects are assigned DBMS labels and so trusted and exempted from OS mandatory controls.
- Sybase adopts this solution by assigning tuple-level labels.

Woods Hole Archs

- General arch is fig 4.5
- A set of untrusted front ends
- It then interface with a monitor (trusted front end) which cannot be bypassed.
- It interfaces an untrusted back end.
Integrity Lock Arch

- Figure 4.6
- Connected via UFE performing pre and post processing of queries.
- An TFE (Trusted filter) is inserted between UFEs and the untrusted DBMS.
- TFE is responsible for enforcing sec functions and multi-level protection.
- Stamp contains sec label and other relevant control data. Stamps are generated and verified by the TFE.
- TFE responsible for generating audit records.
- The problem is leakage of unauthorized info and also inference. \( \Rightarrow \) selection, projection and ... must be handled in TFE or UFE!! Not in the DBMS
- Figure 4.7

Kernelized Arch.

- Figure 4.9
- Trusted OS is used, responsible for the physical access in DB and enforcing mandatory access control.
- DB objects have similar sec labels stored in trusted OS.
- Converting multi-level relations into single level relations access.

Replicated Arch.

- Figure 4.10
- Expensive.
- No implementation in real business.
Research Prototypes

- SeaView
  - SeaView implements the Sea View security model using a kernelized arch.
  - Designed to satisfy the A1 classification of DoD (verification design).
  - Jointly by Oracle, SRI, and Gemini.

Five layers:

1- GEMSOS TCB
   - A Mandatory Security Kernel (GEMSOS security kernel) + the Non-Mandatory TCB.
   - GEMSOS security kernel implements the requirement for mandatory policy.
   - Enforces the mandatory sec policy through a label-based mechanism.
   - Has 8 protection rings:
     - Ring attributes are assigned to each object.
     - A ring number is assigned to each subject.
     - A subject can access an object if its ring number is consistent with the object ring attribute.
   - The Non-Mandatory TCB is responsible for audit and group management.

2- Resource Manager is responsible for providing the requirements of the Oracle DBMS and Sea View.
   - It is a special-purpose OS outside the TCB, since the TCB should have the minimal design.
   - Funcs: Creation of the file system, high-level device drivers, mapping of the high-level objects of DBMS to low-level objects of the OS.

3- Oracle Mandatory Prototype:
   - Management of the single-level objects.
   - Composed of the Oracle Run-Time environment + rewritten Oracle utilities
   - Oracle pre-processor for supporting execution of embedded SQL

4- MSQL processor:
   - Dealing with multi-layer relations.
   - Converting embedded SQL into normal SQL statements.
   - It is a special SQL designed for SeaView, to deal with multilevel relations.

5- SeaView Users layer composed of the functions required to manage the DB that can be left untrusted.
A1 Secure DBMS

- ASD is a multilevel secure relational DBMS designed in 1992.
- A prototype is available.
- Objective: meeting the A1 criterion of DoD classification.
- Developed by the ADA language.
- On top of the A1secure OS.
- Permits interconnection of the trusted and untrusted systems.
- It guarantees that data has been protected via both mandatory and discretionary access control rules.
- Only one copy of data is stored in the system, accessed by different security levels.

ASD

- Can work in 3 different modes
  - As a DBMS server on a LAN
  - As a back-end DBMS for a host computer (single-level and multi-level)
  - As a host-resident DBMS.
- Fig 4-12 (to be drawn on the board)
- Comm between untrusted processes is done through TCB.
- Enforces both BLP and Biba rules.

Commercial Products

- Ingres
- Oracle
- Sybase (Sybase secure server)
- Informix
- SQLBase
- Table 4.2 page 266

Design of Secure DBs

- As the secondary requirement: most cases.
- Security is added as a library.
- As the primary requirement.
- Methodologies for secure software development is normally used in this case.
- Some cases OS security features are utilized.

DoD guidelines

- Methodology for secure DB design:
  - Preliminary analysis
  - Security requirements and policies
  - Conceptual design
  - Logical design
  - Physical design
    - Figure 4.13
  - Separating security policies from security mechanisms.

- Preliminary analysis
  - System risks
  - Features of the database environment: single level or multi-level security support.
  - Applicability of existing security products
  - Integrity of the security products
  - Performance of the resulting security system
Requirement analysis and sec policy selection

• Protection needs for different types of risk are different for different database systems.
• Sensitivity of information.
• High-risk and low-risk systems.
• Data accessibility of who has access to which data.
• Number and types of users.
• Reliability of security on the selected technologies.
• Degree of vulnerability to which the environment is exposed.

Security policy selection

• Secrecy vs integrity vs reliability
• Maximum sharing vs minimum privilege
• Granularity of control
• Attributes used for access control: S/O, location, time, ...
• Integrity
• Priorities
• Privileges
• Authority
• Inheritance

Conceptual design

• Designed through
  – Identification of subjects and objects
  – Identification of access rights granted to Ss over Os.
  – Analysis of the propagation of authorizations in the system through grant/revoke privileges.
• Should be
  – Complete, of all security requirements initially stated
  – Consistent, if there should be no access, should not be directly or indirectly.

Logical and physical design

• Logical design, mapping of the conceptual design into a logical model supported by the specific DBMS being used; e.g. considering views, queries, ...
• Physical design, how to implement access rules and their relationship with the physical structure of DB.

Recommendation of implementing security mechanisms.

• Economy of mechanisms: simplicity as much as possible.
• Efficiency
• Linearity of costs, the operation cost should be proportional to the actual use of the mechanism
• Privilege separation (responsibilities)
• Minimum privilege.
• Complete mediation
• Known design. Sec techniques should be well known for the client.
• Security by default.
• Minimum common mechanisms: Mutual independence between mechanisms.

• Psychological acceptability: easy to use, avoiding heavy restrictions on users to encourage protection, not trying to disable it!!
• Flexibility: having different policies
• Isolation: isolation of security mechanisms from the other components, so more resistant.
• Verifiability
• Completeness and consistency
• Observability: the mechanism and the possible attacks against it should be controllable.
• Problem of residuals: residuals (from the terminated processes) must be erased.
• Invisibility of data: if a user is unauthorized to access data must be unauthorized about the structure of data.
• Work factor: too much effort to break a mechanism
• Intentional traps: Monitor the behavior and the sources of attacks.
• Emergency measures
• Secure hardware
• Programming language, the language, its compiler, …
• Correctness

The person relation schema for illustrating statistical database security
[FROM ELMASRI/NAVATHE FIGURE 22.3]