Introduction to Secure Database

한국산업기술대학교
KOREA POLYTECHNIC UNIVERSITY

오 용 철

Introduction to Database Security

- Importance of Security
- Emergence of Federated Database Management Systems
  - 1970 - 1980 : Relational Data Model
  - 1980 - 1990 : Semantic Data Models, Object-Oriented Model
  - 1990 - present : OO Models, OODBMSs
    Unified Models e.g. UniSQL, Illustra, Oracle8, UDB(IBM)
  - 2000: Autonomous Databases in Federated Environment
- Need of Global Security Model

Database Security

- Definition
  - Database security is concerned with the ability of a computer system to enforce a security policy governing the disclosure, modification, or destruction of information.
- Requirements of the Database Security Problems (broad meaning)
  - secrecy
  - integrity
  - availabilty
- Access Control ensures secrecy
  - Access control mechanism checks the right of the user against a set of authorizations.
  - Authorization states which user can perform which action on which data, and it is stated by the security policies of the organization.
  - Our focus: authorization model and access control mechanism.

Models of Secure Databases

- DAC related models
  - Access Matrix Model *
  - Take-Grant Model
  - Lattice Model
  - ...
- MAC related models
  - Bell-LaPadula Model *
  - Clark-Wilson Model
  - Biba Model
  - ...
- Unified Models

Discretionary Access Control

- The control restricts the access to objects based on the identity of subject and/or group
- The controls are *discretionary* in the sense that a subject with a certain access permission is capable of passing that permission (perhaps indirectly) on to any other subjects
- Can be stated in terms of a set of <subject, object, type of authorization>
- Advantages
  - A well known technique, only few open research issues
  - Most of the common commercial DBMSs support it.

Mandatory Access Control

- The control restricts access to objects based on
  (1) the security of the information (as represented by a level) and
  (2) the formal authorization of subject (clearance level) to access information.
- Security Level
  - Sensitivity of Data (Object)
  - Clearance of Users (Subject)
- Access Rules
  - Read Access; the level of Subject *dominates* the level of Object
  - Write Access: the level of Object *dominates* the level of Subject
- Example of Security Level
  - Unclassified < Classified < Secret < Top Secret

Information Propagation Tree (IPT)

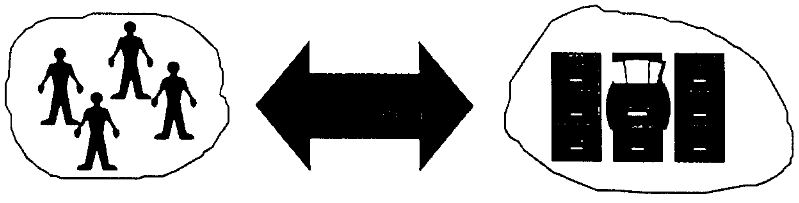
- Definition:
  - Information Propagation Tree (IPT) is a rooted tree which can represent security scheme whose root is the creator of that information, and it has the following diagrammatic convention.
- Diagrammatic Convention
  - IPT = { V, E }
    - V = {terminal node, non-terminal node}, a set of subjects
      - terminal node will be double lined circle.
      - non-terminal node will be single lined circle.
    - E = a set of transactions
    - Arcs represent the authorizations passed on by a user or the security administrator.
      - Arc(a,b) = propagated access rights from a subject "a" to a subject "b"

Information Propagation Tree (IPT)

- All database security policies are viewed as *extended GRANT* rules.
- In DAC, authorizations are determined by GRANT (by definition)
- In MAC, authorizations are also determined by implicit GRANT.
  - We can consider that authorizations are granted by someone according to its domination rules. Therefore, the grantor is trivial in MAC.
  - Every subject is grantee.
- Advantages
  - It is supported by a theoretical model.
  - It explains the difference between DAC and MAC graphically.
  - It works well as a conceptual Model.

SEER: Security Enhanced Entity Relationship model

- Review of E-R model
  - There has been no mention of security features.
- Introduction
  - ***security relationship***: Security mechanism can be explained with a relationship between two groups: subject and object.
  - In DAC, the security relationship is ***access type.***
  - In MAC, the security relationship is ***dominance of classification.***

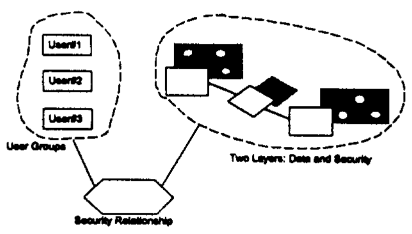


Subjects (Users)

Objects (Data)

An Extended E-R Model

- two layers: one for data, one for modeling security scheme and authorization history
- user group (role): a set of user entities
- security relationship: meta data between two groups



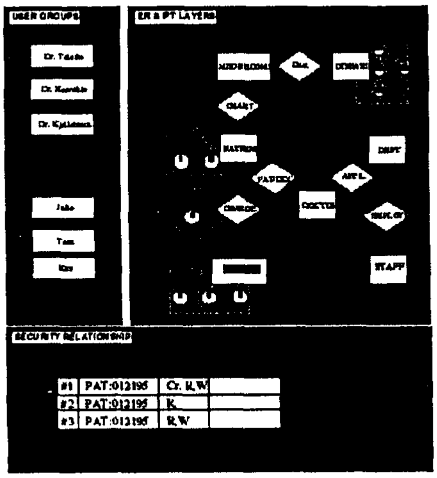
```
user_group(usergroup#, usergroup_name, usergroup_description, ......)

security_relation(usergroup#, data_object_id, authorization_type, site, dominance ...)
```

Developing SEER model from underlying relational databases

- Relational schema analysis.
- Constructing an integrated E-R schema from existing models.
  - Our methodology deals with the use of various keys [ON95],[Oh91] in relations and *reverse* engineers an ER model.
  - Note that this problem has been addressed by several researchers including [Bati91], [Bati86], and [Elma94]. Hence we will not discuss it here.
- Constructing USER GROUPS based on the possible information of active agents.
- In each step of conversion into E-R, checking possible security information and authorization history, and constructing the second layer (security layer with IPT).
  - If the information is not appropriate to these two layers, such as instance level information, then the meta relationship between subjects and objects, called the SECURITY RELATIONSHIP is designed to store this type of information.

Example figure of SEER



#1	PAT:012195	Cr. R,W	
#2	PAT:012195	R	
#3	PAT:012195	R,W	

Advantages of SEER approach

- Handles security schemes and constraints as well as history of authorizations.
  - proposed IPT to capture the history of propagation of access rights among user groups
  - useful for tracking the history of authorizations.
- Works at a conceptual level in the extended ER model.
  - useful for conceptually understanding the security related information regarding the underlying databases.
- Deals with component databases that come from different underlying environment with their own security schemes.
  - useful for considering different security schemes in one framework

What is the problem for storing secure data?

- Multilevel Secure Database (Kernelized Architecture)
- Inference Problems in Index
  - Classified Multiple Indexes
  - Performance Issues (unnessary scannings)
- Crosslinked Index for Range Query
  - Maintenance Problem
- Can we use one index?
  - Better Performance, Better Maintenance, and Secure Access

Type of index for MLSDB

Type of Index	Proposed by	Description
Conventional Index		There is no security protection.
Classified Multiple Index (CMI)	Commission of Air Force Studies Board and National Research Council	This is generally used in multilevel secure databases.
Crossed Linked Index (CLI)	Luef and Pernul	This prevents unnecessary scanning multiple indexes, but requires maintenance of a cross linked data structure.
Trused Common Index (TCI)	Oh and Navathe	This uses one index for multilevel secure database, and allows maintaining of security.

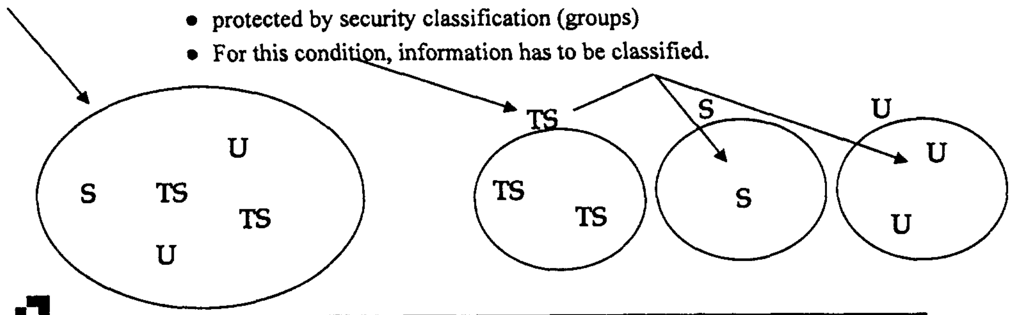
Inference Blocking

- Three types of blocking
  - blocking entire index from the root (mutiple indexes, cross link)
  - blocking the leaf nodes
    - key index entries in internal nodes can still be inferenced
    - how to protect the leaf nodes
  - blocking the data page
- Extendible Hashing for Indexing
  - Fagin R., J. Nievergelt, N. Pippenger, and H. H. Strong, "Extendible hashing - a fast access method for dynamic files," ACM Transaction on Database Systems, 4 (September 1979), pp.315-355
- Inference blocking on data page
  - grouping the data page according to security classification

Security Preservation

Security is preserved, if the system has either of the following conditions.

- **Condition 1:** No sensitive information at all
  - no information or not sensitive information
  - If this condition is satisfied, information can be exposed.
- **Condition 2:** Selective information is present
  - protected by security classification (groups)
  - For this condition, information has to be classified.

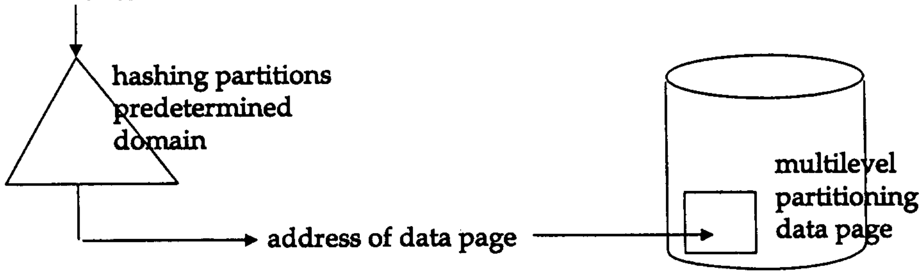


B+ Tree Index with sensitive values

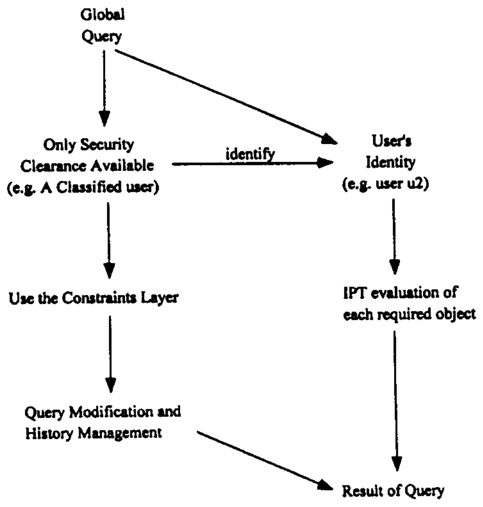
- B+ index is based on comparing key values.
  - index is vulnerable
  - index has sensitive information
  - inference problems from index values
- Condition 1 does not hold; there can be sensitive information.
- Therefore, we use a different index scheme which does not use key value comparison method for building and searching the indexes.
- Extensible hashing uses predetermined domain partitioning method.

Indexing scheme

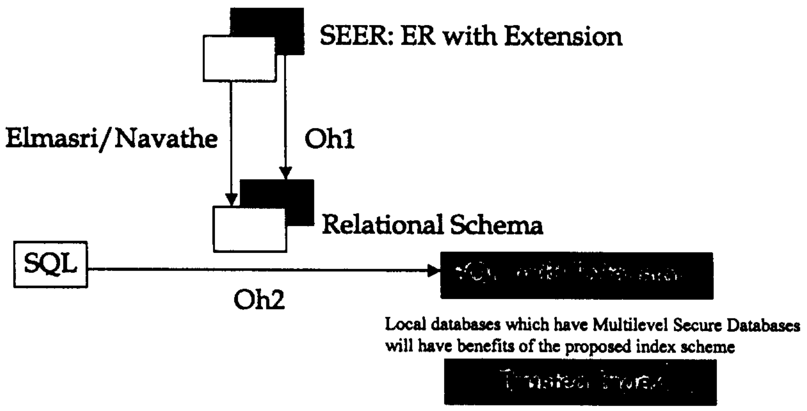
- Condition 2 does not hold, because the index does not have levels of classification.
- We use one index, but disk pages are partitioned or grouped to support classification of data.



Query Operations in SEER



Using Mapping to Relational



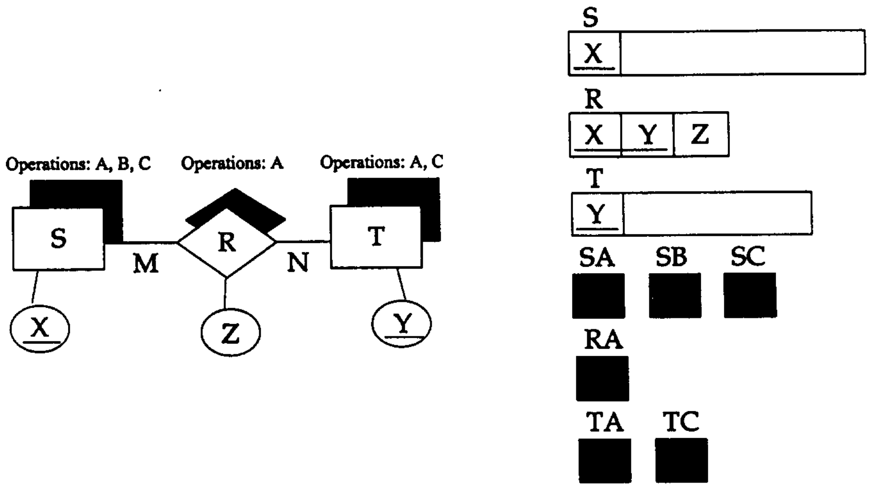
Local databases which have Multilevel Secure Databases will have benefits of the proposed index scheme

ER to Relational translation

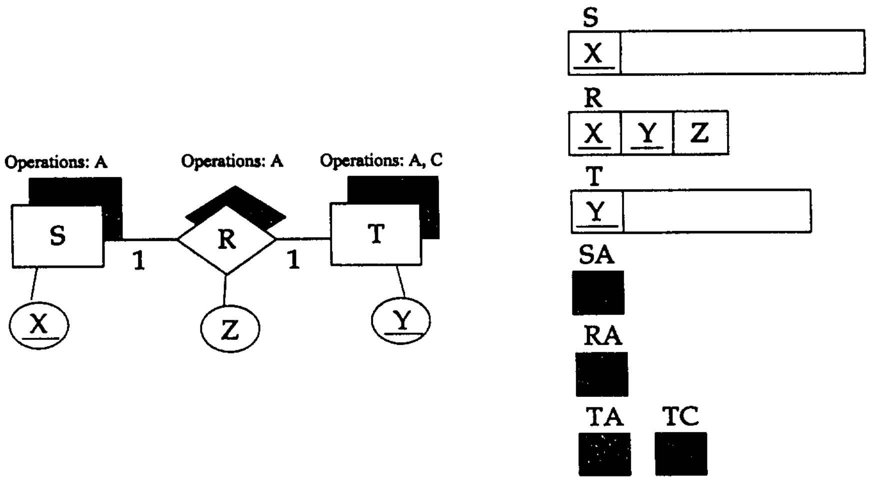
- Entity type to relation
- Relationship type to relation
  - n:1
  - 1:1
  - m:n
- Security information to relation

- weak entity, multivalued attribute, n-ary relationship

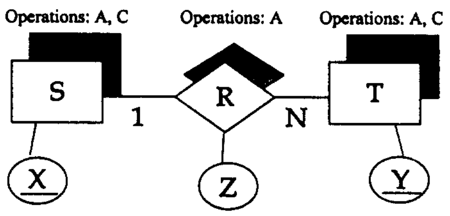
Many-to-Many Relationship

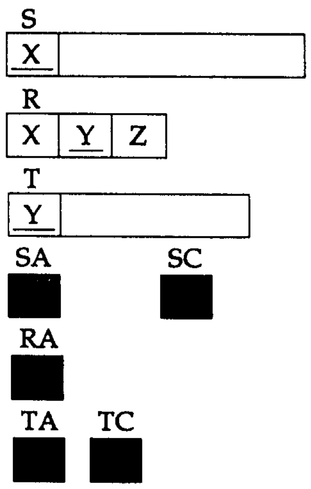


1-to-1 Relationship



1-to-Many Relationship





Query Modification for SEER

SELECT :

```
SELECT {*|Ai[,Aj] . . .}
FROM R
[WHERE p]
```

```
SELECT X,Y                SELECT X,Y
FROM S,T;     ------>     FROM S,T
                          WHERE U IN (SELECT * FROM SA)
                             AND U IN (SELECT * FROM TA);
```

Insert, Delete, and Update

```
INSERT INTO R [(Ai[,Aj] ...)]            ──────►   INSERT INTO R [(Ai[,Aj] ...)]
VALUES (ai[,aj] ...);                              VALUES (ai[,aj] ...)
                                                   WHERE U IN (SELECT * FROM RB);

UPDATE R                                           UPDATE R
SET Ai= Si[,Aj=Sj] ...                   ──────►   SET Ai= Si[,Aj=Sj]
[WHERE p];                                         WHERE [p AND]
                                                       (U IN (SELECT * FROM RC));

                                                   DELETE FROM R
DELETE FROM R                            ──────►   WHERE [p AND]
[WHERE p];                                             (U IN (SELECT * FROM RC));
```

The Constraint based Approach to Query Processing

- Secure Query Processing in DDB (Stachour 1990)
  - Simple Constraint
  - Contents-based Constraint
  - Context-based Constraint
  - Aggregate Constraint
  - Time Constraint
- Security Constraint Processing (Pernul 1992)
  - Simple Constraint
  - Contents-based Constraint
  - Complex Constraint
  - Level-based Constraint
  - Inference-based Constraint

Security Constraints Examples

- **Simple Constraint**
  - Name of Patients is *"Unclassified"*.
- **Context Constraint**
  - PATIENT's NAME will be *"Secret"* if he/she lives in "Atlanta".
- **Content Constraint**
  - NAME and ADDRESS together will be *"Secret"*.

Content-based Constraint

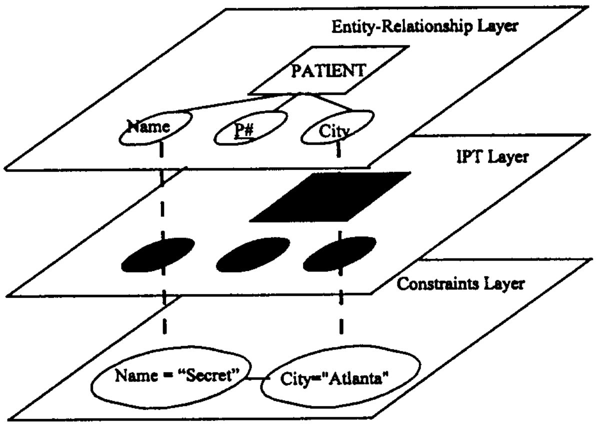
Algebra:

$$\text{Level}(\Pi_{\text{Name}}\ \sigma_{\text{Address}=\text{"Atlanta"}}(\text{Patient})) = \text{"Secret"}$$

Extended SQL:

```
ASSERT SECURITY S1 TO
   SELECT name
   FROM patient
   WHERE address = 'Atlanta'
   LEVEL secret;
```

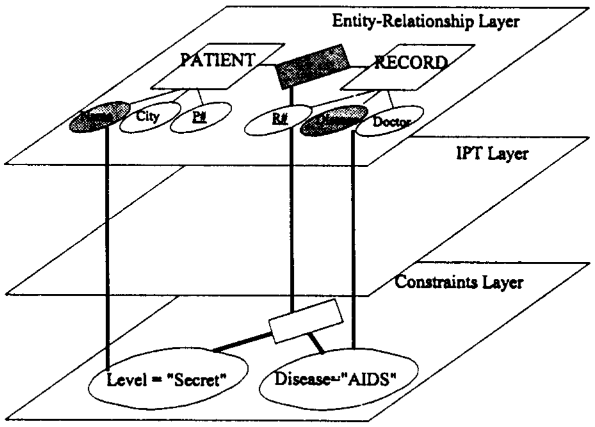
Constraint Layer: Content-based



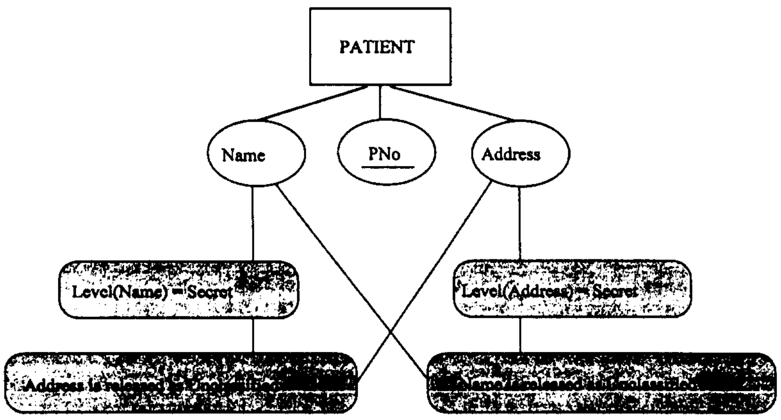
Query Using Constraint layer

- Query : unclassified user wants to retrieve all names.
  - search the constraint node attached name in order to find any security violation (level is greater than unclassified)
  - the constraint node is found: level (name) = "secret"
  - check all other nodes attached that constraint node
  - the constraint node is found: address = "Atlanta"
  - negation of the constraint is what we want.
  - **Result:** retrieve all names (not address in Atlanta).
- Query: Secret user wants to retrieve all names.
  - check the constraint which both violates security constraints and is attached to the Name node
  - Not found.
  - **Result:** no modification

Content-based (join case)



Context-based Constraint


PATIENT
Name
PNo
Address
Level(Name) = Secret
Level(Address) = Secret
Address is released to Unclassified
Name is released to Unclassified

Publications

1. Yong-Chul Oh and Shamkant B. Navathe, "SEER: Security Enhanced Entity Relationship Model for Secure Relational Databases," *International Conference of Object-Oriented Entity Relationship (OOER'95)*, Australia, 1995.

2. Yong-Chul Oh and Shamkant B. Navathe, "Avoiding Inference Problem in Secure Databases Using Page Level Classification," *Security Workshop on Security of Database and Expert Systems Applications (DEXA'98), Austria, 1998.*