Data Security

Dr.Jeevani Goonetillake
UCSC

Introduction to Database Security Issues

- Types of Security
  - Legal and ethical issues
  - Policy issues
  - System-related issues
  - Multiple security levels

Introduction to Database Security Issues

Threats to databases

- Loss of integrity
- Loss of availability
- Loss of confidentiality

Objectives to be considered when designing a secure database application:

- Integrity
- Availability
- Secrecy

Introduction to Database Security Issues

To protect databases against these types of threats four kinds of counter measures can be implemented :

- *access control,*
- *inference control,*
- *flow control and*
- *encryption.*

Database Security and the DBA

- The DBA is the central authority for managing a database system.

- DBA's responsibilities include granting privileges to users who need to use the system and classifying users and data in accordance with the policy of the organisation.

Database Security and the DBA

- DBA-privileged commands include the commands for performing the following types of actions:
  - Account creation
  - Privilege granting
  - Privilege revocation
  - Security level assignment

Access Control

- A DBMS offers two main approaches to access control:
  - **Discretionary access control**

    Govern the access of users to information on the basis of user's identity and predefined discretionary "rules" defined by the security administrator

    The rules specify, for each user and object in the system, the types of access the user is allowed for the object
  - **Mandatory access control**

    Govern the access to the information by the individuals on the basis of the classification of subjects and objects in the system.

Discretionary Access Control

The typical method of enforcing **discretionary access control** in a database system is based on the granting and revoking **privileges**.

Types of Discretionary Privileges

- The *account level*: At this level, the DBA specifies the particular privileges that each account holds independently of the relations in the database.

- The *relation (or table level)*: At this level, the DBA can control the privilege to access each individual relation or view in the database.

Account Level Privileges

The privileges at the **account level** apply to the capabilities provided to the account itself and can include

- CREATE SCHEMA or CREATE TABLE privilege, to create a schema or base relation;
- CREATE VIEW privilege;
- ALTER privilege, to apply schema changes such as adding or removing attributes from relations;
- DROP privilege, to delete relations or views;
- MODIFY privilege, to insert, delete, or update tuples;
- SELECT privilege, to retrieve information from the database by using a SELECT query.

- Not defined as part of SQL2.

Relation Level Privileges

- The second level of privileges applies to the **relation level**, whether they are base relations or virtual (view) relations.

- In SQL the following types of privileges can be granted on each individual relation R:

  – SELECT (retrieval or read) privilege on R : This gives the account, the privilege to use the SELECT statement to retrieve tuples from R.

  – MODIFY privileges on R : This gives the account the capability to modify tuples of R. In SQL this privilege is further divided into UPDATE, DELETE, and INSERT privileges to apply the corresponding SQL command to R.

Relation Level Privileges

- REFERENCES privilege on R: This gives the account the capability to reference relation R when specifying integrity constraints. The privilege can also be restricted to specific attributes of R.

Modify Privilege

- **Insert authorisation**
  - allows insertion of new data, but no modification of data.
  - insertion can be for some of the specified attributes and the remaining will take default or NULL values.

. **Update authorisation**

- allows modification of data but not deletion
- modifications can be for the specified attributes.

Modify Privilege

Delete authorisation

- allows deletion of data

- A user may be assigned all, none or a combination of these types of authorisation.

Discretionary Access Control

- In SQL2, the DBA can assign and owner to a whole schema by creating the schema and associating the appropriate authorization identifier with that schema, using the CREATE SCHEMA command.

- The owner account holder can pass privileges on any of the owned relation to other users by **granting** privileges to their accounts.

Example

Suppose that the DBA creates four accounts -- A1, A2, A3, and A4-- and wants only A1 to be able to create base relations; then the DBA must issue the following GRANT command in SQL:

```
GRANT CREATETAB TO A1;
```

In SQL2 the same effect can be accomplished by having the DBA issue a CREATE SCHEMA command as follows:

```
CREATE SCHEMA EXAMPLE AUTHORIZATION A1;
```

Example

User account A1 can create tables under the schema called EXAMPLE.

Suppose that A1 creates the two base relations EMPLOYEE and DEPARTMENT;

A1 is then **owner** of these two relations and hence *all the relation privileges* on each of them.

Granting Privileges

- GRANT command:
  - Specify privileges for users on database objects.

```
GRANT   <privilege list>
ON        <relation or view>
TO        <user list>
```

```
GRANT     UPDATE(Designation)
ON        Employee
TO        A2, A3
```

```
GRANT     SELECT, INSERT
ON        Employee
TO        A2
```

Revoking Privileges

- REVOKE Command:
  - Remove privileges from users on database objects.

REVOKE <privilege list>
ON <relation or view>
FROM <user list>

Revoking Privileges

REVOKE SELECT
ON Employee
FROM A2

REVOKE UPDATE(Designation)
ON Employee
FROM A2

Revoking Privileges

- The revocation of a privilege from a user may cause other users also to lose that privilege. This behavior is called *cascading* of the revoke.

- The REVOKE statement may also specify RESTRICT. In this case, an error is returned if there are any cascading revokes, and the revoke action is not carried out.

Propagation of Privileges

- Whenever the owner A of a relation R grants a privilege on R to another account B, the privilege can be given to B *with* or *without* the GRANT OPTION.

- If the GRANT OPTION is given B can also grant that privilege on R to other accounts.

GRANT SELECT ON Employee, Department TO B WITH GRANT OPTION;

GRANT SELECT ON Employee TO C;

Example

Suppose that A1 wants to give back to A4 a limited capability to SELECT from the EMPLOYEE relation and wants to allow A4 to be able to propagate the privilege.

The limitation is to retrieve only the NAME, BDATE, and ADDRESS attributes and only for the tuples with DNO=5.

Example

A1 has to create a view:

```
CREATE VIEW A4EMPLOYEE AS
SELECT NAME, BDATE, ADDRESS
FROM EMPLOYEE
WHERE DNO = 5;
```

After the view is created, A1 can grant SELECT on the view A3EMPLOYEE to A4 as follows:

```
GRANT SELECT ON A4EMPLOYEE TO A4
                        WITH GRANT OPTION;
```

Example

Finally, suppose that A1 wants to allow A4 to update only the SALARY attribute of EMPLOYEE;

A1 can issue:

GRANT UPDATE ON EMPLOYEE (SALARY) TO A4;

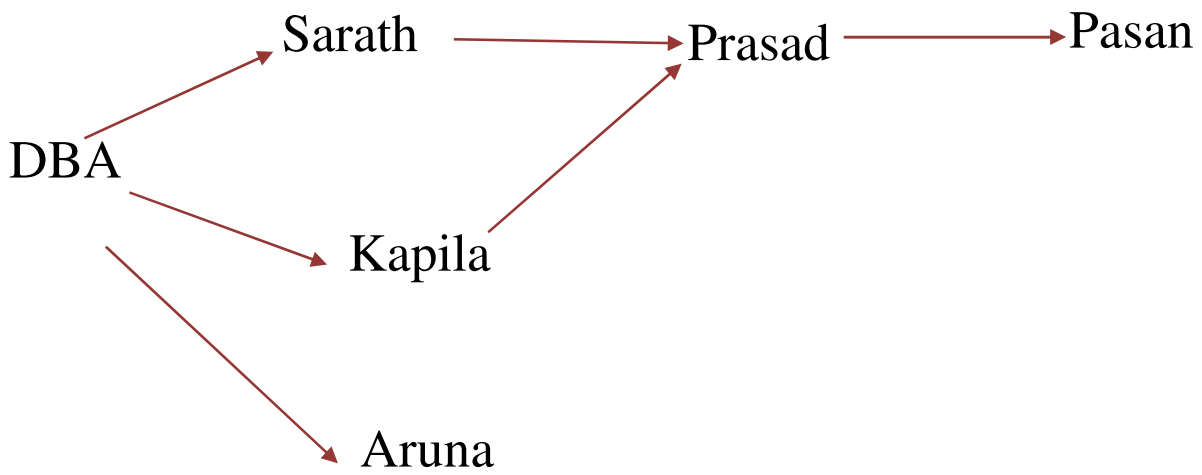
Example

- A bank maintains a database to keep track of customer, branch, account and loan information. corresponding bank relational schema is given below.

  - Branch(BranchName, Code, City, Assets)
  - Account(AcctNo, Balance, Acc-type, BranchName)
  - Loan(LoanNo, Amount, Loan-type, BranchName)
  - Customer(CustomerNo, Name, Address, Phone)
  - C_A(CustomerNo, AcctNo)
  - C_L(CustomerNo, LoanNo)

Example

- Privileges are granted to users of the database as described in the questions given below and as shown in the authorisation diagram.



Example

1. User Sarath is able to retrieve the customer details of each customer at the Colombo branch along with his AcctNo and Balance and is able to update phone and address information of each customer. Write SQL statements to allow this.

2. Kapila is able to retrieve the name of each customer who has a loan but does not have an account at the bank.

Example

3. Prasad is able to retrieve the following:

- AcctNo, the name and the account balance of each customer who has a bank balance greater than Rs: 10,000 at the Colombo branch in such account and
- LoanNo, the name and the loan amount of each customer at the Colombo branch who only has loans but no accounts at the bank.

4. To revoke the privileges of Sarath, the following command is issued

REVOKE SELECT, UPDATE(Address, Phone) ON customer_account FROM Sarath RESTRICT;

Explain the impact of it on the users Prasad and Pasan.

Role-based access control (RBAC)

- Permissions are associated with roles, and users are made members of appropriate roles thereby acquiring the roles' permissions. This greatly simplifies management of permissions.

Role-based access control (RBAC)

- Roles are created for the various job functions in an organization and users are assigned roles based on their responsibilities and qualifications.
- Users can be easily reassigned from one role to another.
- Roles can be granted new permissions as new applications and systems are incorporated, and permissions can be revoked from roles as needed.

Creating a Role

CREATE ROLE product_manager;
CREATE ROLE hr_manager;

- **A layer of security can be added to roles by specifying a password, as in the following example:**

CREATE ROLE overall_manager IDENTIFIED by manager_password;

Granting Privileges to Roles

- **Both system and object privileges can be granted to a role.**
- **A role can be granted to another role.**

```
GRANT SELECT, INSERT, UPDATE, DELETE ON
                        Product TO product_manager;

GRANT CREATE USER TO hr_manager;

GRANT product_manager, hr_manager TO
                                overall_manager;
```

Granting Roles to a User

- **GRANT overall_manager TO A1;**

  To grant a role, you must
  - have been granted the role with the ADMIN OPTION or
  - have been granted the GRANT ANY ROLE system privilege or
  - you must have created the role.

- GRANT **overall_manager** TO A1 WITH ADMIN OPTION;

Revoking Roles

- **Revoking a Role from a User**

  The following statement revokes the role **overall_manager** from the user A1:

  REVOKE **overall_manager** FROM A1;

- **Revoking a System Privilege from a Role**

  The following statement revokes the CREATE TABLE system privilege from the **overall_manager** role:

  REVOKE CREATE TABLE FROM **overall_manager**;

Revoking Roles

- **Revoking a Role from a Role**

  To revoke the role **hr_manager** from the role overall_manager, issue the following statement:

  REVOKE hr_manager FROM overall_manager;

Dropping a Role

- Specify the name of the role to be dropped.

  DROP ROLE hr_manager;

Setting a Role

- A user who has been granted one or more roles has to invoke the SET ROLE command to enable or disable roles for the current user session.
- If the role has a password, you must also specify the password to enable the role by using the IDENTIFIED BY clause.

Setting a Role

- Example 1: To activate the role accountant having a password
  - SET ROLE accountant IDENTIFIED BY acct
- Example 2: To disable all roles granted to you for the current session, issue the following statement:
  - SET ROLE NONE

Default Roles

- It is possible to provide a facility to set up a default list of roles to be activated at the time of user login.
- This facility is enabled through the use of DEFAULT ROLE clause of the ALTER USER command.

Default Roles

- Example : To make accountant role as the default active role for A1
  - ALTER USER A1 DEFAULT ROLE accountant
- Example : To make all authorized roles for A1 part of his default active role set except the auditor role
  - ALTER USER Scott DEFAULT ROLE ALL EXCEPT auditor;
- Example : To remove all roles from A1's default active role set
  - ALTER USER A1 DEFAULT ROLE NONE;

Mandatory Access Control

- This is an all-or-nothing method: A user either has or does not have a certain privilege.
- In many applications, and *additional security policy* is needed that classifies data and users based on security classes.
- This approach as **mandatory access control**, would typically be *combined* with the discretionary access control mechanisms.

Mandatory Access Control

- In many applications, and *additional security policy* is needed that classifies data and users based on security classes.
- This approach as **mandatory access control**, would typically be *combined* with the discretionary access control mechanisms.

Mandatory Access Control

Typical **security classes** are

top secret (TS),

secret (S),

confidential (C), and

unclassified (U),

where TS is the highest level and U the lowest:

$$TS \geq S \geq C \geq U$$

Mandatory Access Control

A **multilevel relation** schema R with n attributes would be represented as

$R(A_1,C_1,A_2,C_2, \ldots, A_n,C_n,TC)$

where each C_i represents the classification attribute associated with attribute A_i.

Mandatory Access Control

(a) EMPLOYEE

Name		Salary		JobPerformance		TC
Smith	U	40000	C	Fair	S	S
Brown	C	80000	S	Good	C	S

(b) EMPLOYEE

Name		Salary		JobPerformance		TC
Smith	U	40000	C	null	C	C
Brown	C	null	C	Good	C	C

(c) EMPLOYEE

Name		Salary		JobPerformance		TC
Smith	U	null	U	null	U	U

(d) EMPLOYEE

Name		Salary		JobPerformance		TC
Smith	U	40000	C	Fair	S	S
Smith	U	40000	C	Excellent	C	C
Brown	C	80000	S	Good	C	S

Mandatory Access Control

A multilevel relation will appear to contain different data to subjects (users) with different clearance levels.

In some cases, it is necessary to store two or more tuples at different classification levels with the same value for the *apparent key*.

This leads to the concept of **polyinstantiation** where several tuples can have the same apparent key value but have different attribute values for users at different classification levels.

Polyinstantiation

- Polyinstantiation is a database technique that allows the database to contain different tuples with the same key but with different classifications.

- Polyinstantiation occurs because of mandatory policy.

Statistical Database Security (Inference Control)

- Statistical databases are used mainly to produce statistics on various populations.
- The database may contain confidential data on individuals, which should be protected from user access.
- Users are permitted to retrieve statistical information on the populations, such as averages, sums, counts, maximums, minimums, and standard deviations.

Statistical Database Security

- A **population** is a set of tuples of a relation (table) that satisfy some selection condition.
- Statistical queries involve applying statistical functions to a population of tuples.

Statistical Database Security

- Statistical users are not allowed to retrieve individual data, such as the income of a specific person. Statistical database security techniques must prohibit the retrieval of individual data.

- This can be achieved by prohibiting queries that retrieve attribute values and by allowing only queries that involve statistical aggregate functions such as COUNT, SUM, MIN, MAX, AVERAGE, and STANDARD DEVIATION. Such queries are sometimes called **statistical queries**.

Statistical Database Security

- It is DBMS's responsibility to ensure confidentiality of information about individuals, while still providing useful statistical summaries of data about those individuals to users.
- Provision of **privacy protection** of users in a statistical database is paramount.
- In some cases it is possible to **infer** the values of individual tuples from a sequence statistical queries. This is particularly true when the conditions result in a population consisting of a small number of tuples.