Database Security – Part 2

Access Control

- **Access Control** refers to any means of controlling access to resources in a database.
- Can be seen as **the combination of authentication and authorization** pl

Authentication vs. Authorization

- **Authentication** is the process by which a system can identify users.
  - Who are the users?
  -

– I
co

- **Authorization** is the process by which a system
access a user (who **is already authenticat**
  - Is a user authorized to access or modify a table?
  - ...

Main Approaches to Access Control

1. **Discretionary access control (DAC)**
   - sers such

2. **Mandatory access control (MAC)**
   - Based on **system-wide polici** individual users.
   - SQL doesn’t support MAC but some

3. **Role-based access control (RBAC)**
   - Based on **roles** (can be used with DAC and MAC).
   - SQL support privileges on roles; many DBMSs support RBAC.

Discretionary Access Control (DAC)

- Called **discretionary** because it allows a subject to grant other subjects pr

- D
  (relations, views, etc.) **on the basis of subjects' privileges**.

- SQL supports DAC through the `GRANT` an
  - `GRANT` gives privileges to users;
  - `REVOKE` takes away privileges from users.

Specifying Privileges - Grant

- The **syntax** of the GRANT command:

```
GRANT privileges ON object TO users [WITH GRANT OPTION]
```

- E

RatingStandard(n_

1. `GRANT SELECT ON SUPPLIER T`
2. `GRANT INSERT, DELETE ON SUPPLIER TO Tom;`
3. `GRANT UPDATE (rating) ON SUPPLIER TO Tom;`
4. `GRANT REFERENCES (no) ON RATINGSTANDARD TO Bob;`

Specifying Privileges - Views

- Views provide **an important mechanism for discretionary authorization**.
- The **syntax** of creating a view:

```
[illegible]
    [illegible]
      FROM table_list
    [WHERE condition]
  [GROUP BY attribute_list [HA [illegible]
  [ORDER BY attribute_list];
```

- Creating a view requires `SELECT` privilege on all relations involved in the view definition.

Specifying Privileges - Views

- **Example:** Consider the relation schema:

SUPPLIER(id, sname, city, rating)

H *y), but*

n

Specifying Privileges - Views

- **Example:** Consider the relation schema:

SUPPLIER(id, sname, city, rating)

H y), but
n

Step 1:
```
CREATE VIEW SUPPLIER-PARI
     SELECT id, sname, city
       FROM SUPPLIER
      WHERE city='Paris';
```

Step 2:
```
GRANT SELECT ON SUPPLIER-PARIS TO Bob
```

Users of this view only see part of SUPPLIER (**horizontal subset** by applying `city='Paris'` and **vertical subset** by excluding `rating`).

Revoking Privileges - Revoke

- The **syntax** of the REVOKE command:

```
R
```

E

SUPPLIER(id, sname

1. `REVOKE INSERT, DELETE ON S`
2. `GRANT SELECT ON SUPPLIER TO Bob;`

    Bob is working on the task ... and done!

    `REVOKE SELECT ON SUPPLIER FROM Bob;`

Delegating Privileges

- Can we pass on privileges to others?

Example: Tom, the owner of SUPPLIER, wants to give Bob the right to grant his SELECT privilege on SUPPLIER to other

```
GRANT SELECT ON SUPPLIER TO B
```

One month later ...

```
REVOKE GRANT OPTION FOR SELECT ON SUPPLIER FROM Bob;
```

Australian National University

Delegating Privileges - Recursive Revocation

- The privileges of an object can be given to a user *with* or *without* the `GRANT OPTION`

- A user can only revoke privileges that he or she has gr
  optional keywords in `REVOKE command`:

  - `CASCADE`: revoking the privilege from a specified user also revokes the privileges from all users who received the privilege from that user.
  - `RESTRICT`: revoking the privilege only from a specified user.

Delegating Privileges - Recursive Revocation

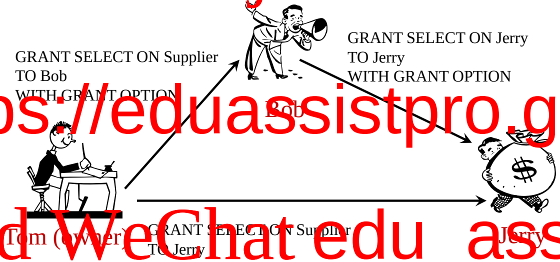
- If a user receives a certain privilege from multiple sources, and the user would lose the privilege only after all sources revoke this privilege.
- **Example:**

```
1
2

3.GRANT SELECT ON SUPPLIER TO Jerry WI

4.REVOKE SELECT ON SUPPLIER FROM Bob C
```

- **Questions:**
  1. Will Bob lose the SELECT privilege on SUPPLIER?
  2. Will Jerry lose the SELECT privilege on SUPPLIER?

Delegating Privileges - Recursive Revocation

- **Example:**



GRANT SELECT ON Supplier
TO Bob
WITH GRANT OPTION

Bob

GRANT SELECT ON Jerry
TO Jerry
WITH GRANT OPTION

Tom (owner)

GRANT SELECT ON Supplier
TO Jerry

Jerry

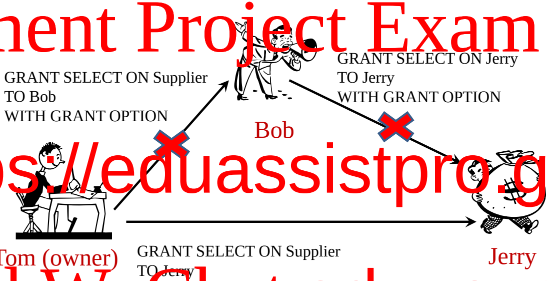
1.GRANT SELECT ON Supplier TO Bob WITH GRANT OPTION; (by Tom)

2.GRANT SELECT ON Supplier TO Jerry; (by Tom)

3.GRANT SELECT ON Supplier TO Jerry WITH GRANT OPTION; (by Bob)

Delegating Privileges - Recursive Revocation

- Example:



GRANT SELECT ON Supplier
TO Bob
WITH GRANT OPTION

GRANT SELECT ON Jerry
TO Jerry
WITH GRANT OPTION

GRANT SELECT ON Supplier
TO Jerry

```
4.REVOKE SELECT ON Supplier FROM Bob C
```

1. Bob will lose the privilege.
2. Jerry won’t lose the privilege.

Delegating Privileges - Propagation

- T
  im

  - Limiting **horizontal propagation**: lim
    `GRANT OPTION` can grant the privilege to a

  - Limiting **vertical propagation**: limits t
    privileges.

Mandatory Access Control (MAC)

- Restrict access to objects based on the **sensitivity of the information** contained in the objects and the formal **authorization** of subjects to access information of such sensitivity.

    -

    - Authorization (e.g., clearances)

Example:

id	sname	city		
1	S1	Paris		
2	S2	Canberra	5	confidential (C)

- Bob with C clearance can only access the second tuple.
- Peter with S clearance can access both tuples.

Role-Based Access Control (RBAC)[1]

- Access rights are grouped by **roles**, and the use of resources is restricted to individuals assigned to specific roles.

[1] Comprehensive Approach to Database Security, Ajoy S. Kumar, 2008