Security and Authorization

Computer Science Department
Columbia University

Introduction to DB Security

- Secrecy: Users should not be able to see things they are not supposed to.
  - E.g., A student can't see other students' grades.
- Integrity: Users should not be able to modify things they are not supposed to.
  - E.g., Only instructors can assign grades.
- Availability: Users should be able to see and modify things they are allowed to.

Access Controls

- A security policy specifies who is authorized to do what.
- A security mechanism allows us to enforce a chosen security policy.
- Two main mechanisms at the DBMS level:
  - Discretionary access control
  - Mandatory access control (not discussed here)

Discretionary Access Control

- Based on the concept of access rights or privileges for objects (tables and views), and mechanisms for giving users privileges (and revoking privileges).
- Creator of a table or a view automatically gets all privileges on it.
  - DMBS keeps track of who subsequently gains and loses privileges, and ensures that only requests from users who have the necessary privileges (at the time the request is issued) are allowed.

GRANT Command

GRANT privileges ON object TO users [WITH GRANT OPTION]

- The following privileges can be specified:
  - SELECT: Can read all columns (including those added later via ALTER TABLE command).
  - INSERT/UPDATE(col-name): Can insert/update tuples with non-null or non-default values in this column.
    - INSERT means same right with respect to all columns.
  - DELETE: Can delete tuples.
  - REFERENCES (col-name): Can define foreign keys (in other tables) that refer to this column.
- If a user has a privilege with the GRANT OPTION, can pass privilege on to other users (with or without passing on the GRANT OPTION).
- Only owner can execute CREATE, ALTER, and DROP.

GRANT and REVOKE of Privileges

- GRANT INSERT, SELECT ON Sailors TO Horatio
  - Horatio can query Sailors or insert tuples into it.
- GRANT DELETE ON Sailors TO Yuppy WITH GRANT OPTION
  - Yuppy can delete tuples, and also authorize others to do so.
- GRANT UPDATE (*rating*) ON Sailors TO Dustin
  - Dustin can update (only) the *rating* field of Sailors tuples.
- GRANT SELECT ON ActiveSailors TO Guppy, Yuppy
  - This does NOT allow the 'uppies to query Sailors directly!
- REVOKE: When a privilege is revoked from X, it is also revoked from all users who got it *solely* from X.

GRANT/REVOKE on Views

- If the creator of a view loses the SELECT privilege on an underlying table, the view is dropped!
- If the creator of a view loses a privilege held with the grant option on an underlying table, (s)he loses the privilege on the view as well; so do users who were granted that privilege on the view!

Views and Security

- Views can be used to present necessary information (or a summary), while hiding details in underlying relation(s).
  - Given ActiveSailors, but not Sailors or Reserves, we can find sailors who have a reservation, but not the *bid*' s of boats that have been reserved.
- Creator of view has a privilege on the view if (s)he has the privilege on all underlying tables.
- Together with GRANT/REVOKE commands, views are a very powerful access control tool.

Authorization Graph

- Authorization Graph: Nodes are users, arcs are privileges passed from one user to the other, labels are the privileges
- The system starts with a "system" node as a "user" granting privileges.
- The rule is that every user's privilege must also be reached by the system node. If not, it has to be removed.
- Cycles are possible

Role-Based Authorization

- In SQL-92, privileges are actually assigned to authorization ids, which can denote a single user or a group of users.
- In SQL:1999 (and in many current systems), privileges are assigned to roles.
  - Roles can then be granted to users and to other roles.
  - Reflects how real organizations work.
  - Illustrates how standards often catch up with "de facto" standards embodied in popular systems.

Security to the Level of a Field!

- Can create a view that only returns one field of one tuple. (How?)
- Then grant access to that view accordingly.
- Allows for *arbitrary* granularity of control, *but*:
  - Clumsy to specify, though this can be hidden under a good UI
  - Performance is unacceptable if we need to define field-granularity access frequently. (Too many view creations and look-ups.)

Statistical DB Security

- Statistical DB: Contains information about individuals, but allows only aggregate queries (e.g., average age, rather than Joe's age).
- New problem: It may be possible to infer some secret information!
  - E.g., If I know Joe is the oldest sailor, I can ask "How many sailors are older than X?" for different values of X until I get the answer 1; this allows me to infer Joe's age.
- Idea: Insist that each query must involve at least N rows, for some N. Will this work? (No!)

Why Minimum N is Not Enough

- By asking "How many sailors older than X?" until the system rejects the query, can identify a set of N sailors, including Joe, that are older than X; let X=55 at this point.
- Next, ask "What is the sum of ages of sailors older than X?" Let result be S1.
- Next, ask "What is sum of ages of sailors other than Joe who are older than X, plus my age?" Let result be S2.
- S1-S2 is Joe's age!

Summary

- Three main security objectives: secrecy, integrity, availability.
- DB admin is responsible for overall security.
  - Designs security policy, maintains an audit trail, or history of users' accesses to DB.
- Statistical DBs try to protect individual data by supporting only aggregate queries, but often, individual information can be inferred.