CSC 405
Introduction to Computer Security

Topic 6. Database Security

Agenda

• Discretionary access control in DBMS
• Mandatory access control and multi-level databases
• Database inference control
Topic 6.1 DAC in DBMS
Outline

- Relational model
- Grant and revoke
- Extension to the basic model
- Questions/comments in reviews

Basic Relational Concepts

- Data is organized as a collection of tables, called RELATIONS
  - Example: two relations - EMP, DEPT
  - EMP: name, title, department
  - DEPT: department, location
- Each row (or record) of a relation is called a TUPLE
- Each relation has a unique name
- Each attribute has a unique name within a relation
- All values in a relation are atomic (indecomposable)
  - As a consequence, we have two tuples for a user
Examples

<table>
<thead>
<tr>
<th>EMP</th>
<th>Name</th>
<th>Title</th>
<th>Dept</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tom</td>
<td>Prof</td>
<td>ECE</td>
<td></td>
</tr>
<tr>
<td>Tom</td>
<td>Prof</td>
<td>CS</td>
<td></td>
</tr>
<tr>
<td>Adams</td>
<td>Prof</td>
<td>ECE</td>
<td></td>
</tr>
<tr>
<td>Smith</td>
<td>Inst</td>
<td>CS</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>DEPT</th>
<th>Name</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS</td>
<td>With Hall</td>
<td></td>
</tr>
<tr>
<td>ECE</td>
<td>Daniels Hall</td>
<td></td>
</tr>
<tr>
<td>Math</td>
<td>Harrelson Hall</td>
<td></td>
</tr>
</tbody>
</table>

Relation Schemes

- A relational database consists of 2 relation schemes:
  - EMP(Name, Title, Dept)
  - DEPT(Name, Location)
- Schemes: structure of the database
- Structured Query Language (SQL)
- SQL "data definition" statements are used to create relations

```
CREATE TABLE EMP
(Name CHAR(15) NOT NULL,
Title CHAR(4),
Dept CHAR(10),
PRIMARY KEY (Name))

CREATE TABLE DEPT
(Name CHAR(10) NOT NULL,
Location CHAR(15),
PRIMARY KEY (Name))
```
SQL

• The SELECT statement

```sql
SELECT Name
FROM EMP
WHERE Dept = 'ECE'
```

• Joins

```sql
SELECT *
FROM EMP, DEPT
WHERE EMP.Dept = DEPT.Name
AND Dept.Location = 'Wither Hall'
```

Views

```sql
CREATE VIEW EMP_LOCATION
AS SELECT Name, Dept, Location
FROM EMP, DEPT
WHERE EMP.Dept = DEPT.Name
```

<table>
<thead>
<tr>
<th>EMP_LOCATION</th>
<th>Name</th>
<th>Dept</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tom</td>
<td>ECE</td>
<td>Daniels Hall</td>
<td></td>
</tr>
<tr>
<td>Tom</td>
<td>CS</td>
<td>Wither Hall</td>
<td></td>
</tr>
<tr>
<td>Abrams</td>
<td>ECE</td>
<td>Daniels Bldg</td>
<td></td>
</tr>
<tr>
<td>Smith</td>
<td>CS</td>
<td>Wither Hall</td>
<td></td>
</tr>
</tbody>
</table>

• Views are "virtual" relations. They can be used to customize relations and to provide security
Discretionary Access Controls

• Decentralized administration
  – Users can protect what they own
  – The owner may grant access to others
  – The owner may define the type of access (read/write/execute) given to others

Access Control Mechanisms

• Identification and Authentication (I&A)
• Security through Views
• Stored Procedures
• Grant and Revoke
• Query Modification
Identification and Authentication

- Identification provided by DBMS can be distinct from that provided by the underlying OS
  - Example: MS SQL server
- Two options
  - I&A through the OS
  - Separate I&A

Security Through Views

<table>
<thead>
<tr>
<th>EMP</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>NAME</td>
<td>DEPT</td>
</tr>
<tr>
<td>Smith</td>
<td>Toy</td>
</tr>
<tr>
<td>Jones</td>
<td>Toy</td>
</tr>
<tr>
<td>Baker</td>
<td>Admin</td>
</tr>
<tr>
<td>Adams</td>
<td>Candy</td>
</tr>
<tr>
<td>Harding</td>
<td>Admin</td>
</tr>
</tbody>
</table>

Users are allowed to access partial information (such as the Toy dept data), but not the detailed information.
Example

CREATE VIEW TOY_DEPT
AS SELECT NAME, SALARY, MANAGER
FROM EMP
WHERE DEPT = 'Toy'

<table>
<thead>
<tr>
<th>TOY_DEPT</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAME</td>
</tr>
<tr>
<td>Smith</td>
</tr>
<tr>
<td>Jones</td>
</tr>
</tbody>
</table>

Example

CREATE VIEW TOY_EMP_MGR
AS SELECT EMP, MANAGER
FROM EMP
WHERE DEPT = 'Toy'

<table>
<thead>
<tr>
<th>TOY_EMP_MGR</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAME</td>
</tr>
<tr>
<td>Smith</td>
</tr>
<tr>
<td>Jones</td>
</tr>
</tbody>
</table>
Example

```
CREATE VIEW AVSAL(DEPT, AVG)
AS SELECT DEPT, AVG(SALARY)
FROM EMP
GROUP BY DEPT
```

<table>
<thead>
<tr>
<th>DEPT</th>
<th>AVG</th>
</tr>
</thead>
<tbody>
<tr>
<td>TOY</td>
<td>12,500</td>
</tr>
<tr>
<td>CANDY</td>
<td>20,000</td>
</tr>
<tr>
<td>ADMIN</td>
<td>45,000</td>
</tr>
</tbody>
</table>

Stored Procedures

- Right to execute compiled programs
- GRANT RUN ON program_A TO ADAMS
- Suppose program_A needs to access the relation EMP. Adams can execute program_A even though he does not have permission to access EMP
Query Modification

- Adams:
  ```sql
  GRANT SELECT ON EMP TO THOMAS WHERE SALARY < 15000
  ```
- THOMAS:
  ```sql
  SELECT *
  FROM EMP
  ```
- DBMS:
  ```sql
  SELECT *
  FROM EMP
  WHERE SALARY < 15000
  ```

The Grant Command

- **GRANT <privilege> ON <relation> TO <users> [WITH GRANT OPTION]**
  - GRANT SELECT ON EMP TO ADAMS
  - GRANT SELECT ON EMP TO ADAMS WITH GRANT OPTION
  - GRANT SELECT, UPDATE(SALARY) ON EMP TO JIM, JILL

- Applied to base relations as well as views
The Revoke Command

- `REVOKE <privileges> [ON <relations>] FROM <users>
  - `REVOKE SELECT ON EMP FROM TOM`
  - `REVOKE UPDATE ON EMP FROM SMITH`
  - `REVOKE RESOURCE FROM ABRAMS`
  - `REVOKE DBA FROM SMITH`

Semantics of Revoke

- A sequence of grant command follow by a revoke operation
  - \( G_1, G_2, \ldots, G_n, R_h \)

- Semantics
  - Equivalent to: \( G_1, G_2, \ldots G_{h-1}, G_{h+1}, G_n \)
Time-stamped Authorizations

Cascading Revocation

Grant sequence:

B revokes privilege from C:

© Sushil Jajodia 2002
Timestamps Make a Difference

A

B

10g

30g

E

D

C

20g

10g

10g

30g

F

D

C

40g

50g

40g

60g
Further Extension

• Make cascading optional

• Permit negative authorizations

The Revoke Command

• REVOKE <privileges> [ON <relations>] FROM <users> [CASCADE]
  – REVOKE SELECT ON EMP FROM TOM
  – REVOKE UPDATE ON EMP FROM SMITH CASCADE
  – REVOKE RESOURCE FROM ADAMS
  – REVOKE DBA FROM SMITH CASCADE
Non-cascading Revocation

A → B → C → D

A → B → D

Why Non-cascading Revoke

• Reasons for revoke
  – Task is done. No need to have the privilege anymore
  – Task is still in progress. But a member left the project (e.g., promoted)
Example

After cascading revocation
Example

![Example Diagram]

After non-cascading revocation

Why Positive & Negative Authorization

- Closed world policy
  - Cannot access unless explicitly granted the right

- Negative authorization
  - User A should not be allowed to read table Emp
  - Need explicit deny policies
Positive & Negative Authorizations

Complication

- It is possible to have two authorizations
  - Grant A privilege p
  - Deny A privilege p

- Negative authorizations override positive authorizations
Problem 1

User B gives D negative authorization at time 50:

In our model, positive authorization granted by A to D becomes blocked, but we do not delete the authorization.

Problem 2

Suppose D receives negative authorization from B at time 60:

What about the privilege given to F by D? Under our approach, it becomes blocked, but we do not delete it.
Revocation When Negative Authorizations Are Present

Given:

A  20

B  40 —

C  10 + g

D  30 + g

E  60 —

F  50 +

Suppose A revokes B’s privilege.

Cascading Revocation When Negative Authorizations Are Present

A  20 —

30 + g

D  50 +

C  60 —

F  50 +
Non-cascading Revocation When Negative Authorizations Are Present

A --- 60 --- B

A --- 30 + g --- C

D --- 50 + --- F

E --- 40 ---