

Database Security

John Sieg, UMass Lowell

SQL Security

- Discretionary - privileges are granted:
GRANT <privilege_list>
ON <object>
TO <user_list>
[WITH GRANT OPTION]
- Privileges are SELECT, INSERT(col), UPDATE(col), DELETE, REFERENCES(col)

John Sieg, UMass Lowell

SQL Example 1

```
GRANT SELECT
ON Student
To Larry
WITH GRANT OPTION
```

John Sieg, UMass Lowell

SQL Example 2

```
GRANT INSERT
ON Student (Sid),
Student(Sname)
To Larry
```

John Sieg, UMass Lowell

SQL Revoke Statement

```
REVOKE [GRANT OPTION FOR]
<privilege_list> ON
  <object>
FROM <user_list>
[CASCADE|RESTRICT]
```

John Sieg, UMass Lowell

SQL Example 3

```
REVOKE SELECT
ON Student
FROM Larry
CASCADE
```

If Larry granted SELECT on Student to Moe, Moe also loses the privilege (usually).

John Sieg, UMass Lowell

SQL Example 4

```
REVOKE SELECT
ON Student
FROM Larry
RESTRICT
```

If Larry granted SELECT on Student to Moe, command is rejected (usually).

John Sieg, UMass Lowell

SQL Example 5

```
Larry:
GRANT REFERENCES
ON Student(Sid)
To Moe
```

- Moe can create a relation schema with a foreign key referencing Sid in Student (not possible with just SELECT privilege).
- Moe can prevent Larry from deleting Student tuples (NO ACTION referential integrity).

John Sieg, UMass Lowell

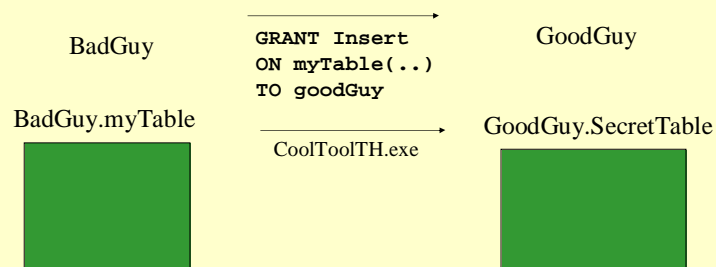
Roles

- Introduced in the SQL:1999 standard, but implemented in many database systems before that.
- Privileges can be granted and revoked to roles.
- Users can be granted and revoked roles, receiving the privileges of the roles.
- The role can be granted to and revoked from, affecting all users granted that role.

Role Example

```
CREATE ROLE dbstudent;  
GRANT select ON student TO dbstudent;  
GRANT insert ON student(sname) TO  
dbstudent;  
GRANT dbstudent TO curly, shemp, moe;  
REVOKE insert ON student(sname) FROM  
dbstudent;  
(All users in dbstudent role lose INSERT  
privilege on student.)
```

Discretionary Security Subject to Trojan Horses



When GoodGuy runs CoolToolTH.exe, it inserts GoodGuy.SecretTable tuples into BadGuy.myTable.

Mandatory Access Control

- An Alternative to Discretionary Access Control, which is subject to Trojan Horses, et al.
- Not Supported in SQL
- The Bell-LaPadula Model has security classes: $TS > S > C > U$

John Sieg, UMass Lowell

Mandatory Access Control, cont'd

- Subject S can read object O only if
 $\text{class}(S) \geq \text{class}(O)$
- Subject S can write object O only if
 $\text{class}(S) \leq \text{class}(O)$

John Sieg, UMass Lowell

Statistical Databases

- Idea: Allow collecting statistics over private data without access to individual records.
- Allow

```
SELECT avg(GPA) FROM STUDENT
```
- Do not allow

```
SELECT avg(GPA) FROM STUDENT  
WHERE sname = "Larry"
```

John Sieg, UMass Lowell

Statistical Databases

- So disallow queries that operate on one record.
- But:

```
SELECT count(*) FROM STUDENT;  
SELECT count(*) FROM STUDENT  
  WHERE sname <> "Larry";  
SELECT sum(GPA) FROM STUDENT;  
SELECT sum(GPA) FROM STUDENT  
  WHERE sname <> "Larry";
```

John Sieg, UMass Lowell

Statistical Databases

- So disallow queries that operate on entire relations
- But:

```
SELECT count(*) FROM STUDENT WHERE SEX = "f";  
SELECT count(*) FROM STUDENT  
  WHERE sname = "Larry" OR SEX = "f";  
SELECT sum(GPA) FROM STUDENT  
  WHERE SEX = "f";  
SELECT sum(GPA) FROM STUDENT  
  WHERE sname = "Larry" OR SEX = "f";
```

John Sieg, UMass Lowell

Public Key Encryption

- P a public key, p a private key, E an encryption function such that
- $E(p, E(P, M)) = M$
- $E(P, E(p, M)) = M$

John Sieg, UMass Lowell

Encrypting

- JS is my public key, js my private key
- B is Bill's public key, b his private key
- I encrypt message M for Bill: $E(B, M)$ and send result M' to Bill.
- Bill decrypts: $E(b, M') = E(b, E(B, M)) = M$

John Sieg, UMass Lowell

Signing

- But Bill may wonder: Perhaps Ken sent this message, not me.
- So instead I send the message:
 $E(B, (E(js, M) \oplus \text{"It's me, your pal John."}))$
- Bill applies b :
 $E(b, E(B, (E(js, M) \oplus \text{"It's me, your pal John."}))) = E(js, M) \oplus \text{"It's me, your pal John."}$
- Bill then applies JS : $E(JS, E(js, M)) = M$

John Sieg, UMass Lowell

Conclusions

- Security enforcement often compromises access (and security!)
- Good security mechanisms are available, but are usually not taken seriously.
- Political concerns (encryption concerns in USA) and commercial concerns (software registration by Microsoft, anti-copy rootkits by Sony) defeat some attempts at security.