# SECURITY

CHAPTER 24 (6/E)

CHAPTER 23 (5/E)

# LECTURE OUTLINE

- Threats and countermeasures
- Access control mechanisms
- SQL’s grant and revoke
- Role of views

# THREATS

- What are the threats?
  - Loss of integrity
    - E.g. Student changing grades for a class they're taking
  - Loss of confidentiality
    - Learning something from the database you shouldn't know
  - Loss of availability
    - "Denial of service"
    - Causing DB to be unavailable to authorized programs / people
- Who's trying to mess with us?
  - Outsiders
    - Amateurs, "Script kiddies", Crackers
    - Corporate competitors
    - Organized crime
    - Government "cyberwarriors"
    - Terrorists / activists
  - Insiders
    - Disgruntled, bribed, or naïve employees
  - Accidental mis-use

# HOW SECURE SHOULD WE MAKE DB?

- **Principle of Easiest Penetration**
  - “A system is only as strong as its weakest link"
    - Perpetrator will attack most vulnerable part(s) of the system
  - To build secure systems, need to think like an attacker!
    - How would you get private information from the Canada Revenue Agency database? Chapters? Facebook? UW?
- **Principle of Adequate Protection**
  - “Security is economics"
  - Don't spend $100,000 to protect a system if maximum loss is only $1000 in damage.
  - Don't spend only $100 to protect a system if maximum loss is $100,000 in damage.

# DEFENSES AGAINST SECURITY BREACH

(Compare to securing your bicycle)

- Prevent it
  - Stop the attack for happening
- Deter it
  - Make the attack harder or more expensive
- Deflect it
  - Make yourself less attractive to attacker
- Detect it
  - Notice that attack is occurring (or has occurred)
- Recover from it
  - Mitigate the effects of the attack

# ASPECTS OF DB SECURITY

- Legal and ethical compliance / Business rules
  - Requirements to maintain accurate information
  - Requirements to disclose information to appropriate people
  - Requirements to *not* disclose information to *in*appropriate people
- Where will security be enforced?
  - by the physical environment?
    - by locked doors? by armed guards?
  - by the hardware?
  - by the software?
    - by the OS? by the DBMS? by applications programs?
    - DBMS includes **security subsystem**
- Levels of security
  - Access / no access
  - Partial access
    - Limited authorizations
    - Authorizations based on user role, time of day, location, etc.
  - Emergency access

# COUNTERMEASURES

- Access control
  - Limiting access to the database (or parts of the database)
  - Requires **authentication** (e.g., through login and password)
  - Usually includes auditing
- Inference control
  - Preventing deductions about database content
  - Access to summary data without ability to determine individuals' data
- Flow control
  - Keeping information from being transferred illegitimately
  - Control over **covert channels**
- Encryption
  - Making information unintelligible unless authorized
  - Making changes traceable to source
  - Requires security keys and key maintenance

# AUDIT TRAIL

- Record all operations on DB
  - Which user, which operation, which data
  - Could be integrated with transaction log
    - Include reads as well as writes
- Audit the log if suspicions arise
  - Review accesses and updates during time period
  - Test for irregular behaviour
- Examples:
  - Identify who last changed record
  - Identify authorized, but unethical reading of confidential data
    - e.g., Britney Spears' medical record

# ACCESS CONTROL MECHANISMS

- **Discretionary** access control (DAC)
  - Granting specific user access to specific piece of data in specific way
  - E.g., “let John Smith insert employees into Employee table”
- **Mandatory** access control (MAC)
  - User’s security clearance must match data’s security class
  - Bell-LaPadula Model
    - No read-up (to protect data)
      - e.g., must have sufficiently high clearance to read *top secret* data
    - No write-down (for flow control)
      - e.g., person with high clearance cannot update unclassified object
- **Role based** access control (RBAC)
  - Users assigned roles
  - Roles entitled to specific permissions on specific data
  - E.g., “emergency physician can update any patient record”

# GRANT AND REVOKE

- DAC support in SQL
- If A1 wants to allow A4 to update only the salary attribute of Employee, A1 can issue
  - GRANT UPDATE ON Employee (salary) TO A4;

  or
  - GRANT UPDATE ON Employee (salary) TO A4 WITH GRANT OPTION;
- Similarly to undo an earlier grant, A1 can issue
  - REVOKE SELECT ON Employee FROM A3;
    - A3 can no longer read Employee
      - unless also granted by other user
    - Revocation also propagates to other users granted privilege by A3

# GRANULARITY OF PRIVILEGES

- Object
  - Table (or view) vs. column
  - SELECT, INSERT, DELETE, and ALTER are not column specific
  - UPDATE and REFERENCES privileges can specify columns
  - SQL does not support tuple-specific privileges
- System
  - Create, alter, drop tables, views, etc.
  - Creator of object gets all (object) permissions on that object

# DAC MODEL: ACCESS CONTROL MATRIX

- Rows represent **subjects** (users, accounts, programs)
- Columns represent **objects** (relations, records, columns, views, operations).
- *M*(*i*,*j*) represents privileges that subject *i* holds on object *j*.
  - Include who granted the privilege (to support revocation)

E.g., privileges ⊆ {select, insert, delete, update, …}, **bold** ⟹ grant option

| | Employee | Department | Dept_locations | Project | Works_on | Dependent |
|---|---|---|---|---|---|---|
| Ashley | **sidu…** (sys) | **sidu…** (sys) | **sidu…** (sys) | **sidu…** (sys) | **sidu…** (sys) | |
| Bobbie | s (Ashley) | s (Ashley) | s (Ashley, Eddie)<br>**idu** (Ashley) | | | **sidu…** (sys) |
| Charlie | **s** (Ashley) | | | **s** (Ashley) | **s** (Ashley) | |
| Dana | s (Ashley, Charlie) | | | | | **s**iu (Bobbie) |
| Eddie | **s** (Ashley) | siu (Ashley) | **s**iu (Ashley) | | | |
| Lee | s (Eddie, Charlie) | | | | | s (Dana) |

# VIEWS FOR SECURITY

- View selects some rows and columns from one or more tables.
  - Other data values are inaccessible through view.
  - E.g., CREATE VIEW SalesStaff AS (
    SELECT Fname, Lname, Address
    FROM Employee
    WHERE Dno IN (SELECT Dnumber
    FROM Department
    WHERE Dname = 'sales')
    AND salary < 50000);
- Grant privileges on view without granting privileges on base tables.

  GRANT SELECT ON SalesStaff TO Smith;
  - Can only access data in view
    - Similar for insert, delete, update
  - Other base data is protected

# DATA INFERENCE

- Derivation of sensitive data from (supposedly) non-sensitive data
- Direct attack
  - Attacker issues query that directly yields sensitive data
  - Might obfuscate query to fool DBMS

```
SELECT AVG(salary)
FROM staff
WHERE lastname = 'Adams'
      OR (sex != 'M' AND sex != 'F')
```

  - Solution: ***k*-anonymity**
    - Every statistical answer must depend on at least *k* records
- Indirect attack
  - Infer sensitive data from statistical results
  - Judicial use of several k-anonymous queries
    - SELECT SUM(salary)
    - SELECT SUM(salary) WHERE lastname != 'Adams'

# LECTURE SUMMARY

- Overview of database security
  - Threats and countermeasures
- Discretionary Access Control
  - SQL's grant and revoke
  - Security through views
- *k*-anonymity