Chapter 5 Database Security
(Combines the information from Textbook, Slides and lecture)

5.1 Database

- Structured collection of data stored for use by one or more applications (like excel spreadsheet, Oracle)
- Contains the relationships between data items and groups of data items
- Query language: Provides a uniform interface to the database
- Primary Key: each row has a primary key, used to uniquely identify each row in a table
- Foreign key: When relating two tables, the attribute that define the primary key in one table must appear as attributes (Foreign key) in another table.

5.2 Database management system (DBMS) see slides

5.4 Structured query language & attacks

- SQL
  - Standardized language to define schema, manipulate, and query data in a relational database
  - the same basic syntax and semantics:
    - Create tables
    - Insert and delete data in tables
    - Create views: often used to limit what you can see from the database
    - Retrieve data with query statements
- SQL injection attacks
  - attacks that exploit a security vulnerability occurring in the database layer of an application (such as queries).
    - Sends malicious SQL commands to the database server
      Like: SELECT rows FROM student WHERE name='Russ'; DROP table student;
Most common attack goal is bulk extraction of data

Like: SELECT rows FROM student WHERE name='Russ'; SELECT * FROM student;

- works by prematurely terminating a text string and appending a new command
- the attacker terminates the injected string with a comment mark "--". Subsequent text is ignored at execution time.

SQLi attack avenues

- User input: like form submission from web application
- Server variables: placing data into HTTP and network headers
- Second-order injection: rely on data already present in database to trigger an SQL injection attack
- Cookies: alter cookies such that when the application server builds an SQL query based on the cookie’s content, the structure and function of the query is modified
- Physical user input: conventional barcodes, RFID tags, etc

SQLi countermeasures:

- Defensive coding:
  - Defensive coding practices: like checking user input
  - parameterized query insertion: pass value parameters separately, use prepared statements
  - SQL DOM: enables automated data type validation and escaping
- Detection: signature based, anomaly based, code analysis
- Run-time prevention
  - Check queries at runtime to see if they conform to a model of expected queries

5.5 Database access control

- DBMS can support a range of administrative policies
  - Centralized administration: A small number of privileged users may grant and revoke access rights.
- Ownership-based administration: The owner (creator) of a table may grant and revoke access rights to the table.
- Decentralized administration: In addition to granting and revoking access rights to a table, the owner of the table may grant and revoke authorization rights to other users, allowing them to grant and revoke access rights to the table. (more like Discretionary Access control)

- Access control in SQL
  - Grant one or more access rights or can be used to assign a user to a role
    - GRANT { privileges | role } [ON table] TO { user | role | PUBLIC }
      [IDENTIFIED BY password] [WITH GRANT OPTION]
  - Revoke access rights
    - REVOKE { privileges | role } [ON table] FROM { user | role | PUBLIC }
  - Access rights:
    - Select, Insert, Update, Delete, References

- Often use RBAC(role-based access control), like Canvas.
  - Classify users in 3 categories (in DAC)
    - Application owner: An end user who owns database objects (tables, columns,
    - End user: An end user who operates on database
    - Administrator: User who has administrative responsibility for part or all of the database.
  - A database RBAC facility needs to provide the following capabilities:
    - Create and delete roles.
    - Define permissions for a role.
    - Assign and cancel assignment of users to roles.

5.6 Inferential attack
- Performing authorized queries and deducing unauthorized information from the legitimate responses received.
- reconstruct the information by sending particular requests and observing the resulting behavior of the Website/database server
- Blind SQL injection
Detection:
- Inference detection during database design: like altering the database structure, using more fine-grained access control (may reduce availability)
- Inference detection at query time: If an inference channel is detected, the query is denied or altered (may also hinder legit user query).

5.7 Database encryption
- Encryption becomes the last line of defense in database security, besides firewall, authentication…
- Two disadvantages:
  - Key management: providing secure keys to a wide range of users
  - Inflexibility: difficult to perform record searching, sorting
- Encryption can be applied to the entire database, at the record level, at the attribute level (encrypt selected columns), or at the level of the individual field.
- Work with database in its encrypted form: (see fig 5.9)
  - Four entities:
    - Data owner
    - User: that presents requests (queries) to the system.
    - Client: Frontend that transforms user queries into queries on the encrypted data stored on the server.
    - Server: receives the encrypted data from owner and makes them available for distribution to clients.

Reference: