Chapter 5
Database and Cloud Security

Databases
- Structured collection of data stored for use by one or more applications
- Contains the relationships between data items and groups of data items
- Can sometimes contain sensitive data that needs to be secured
Query language
  - Provides a uniform interface to the database

Database management system (DBMS)
- Suite of programs for constructing and maintaining the database
- Offers ad hoc query facilities to multiple users and applications

Packetized network can overcommit the channel
IP -> No guarantees
TCP -> Connection based
Databases use RBAC
Primary key identifies an entry in a table
Queries get sent across packetized network
-> Poses security problems

SQL injection attack -> insert "i;"
followed by new statement
Prepared statements -> used to prevent against SQL injections

Query Language -> Access to database
Structured Query Language (SQL)
- Standardized language to define schema, manipulate, and query data in a relational database
- Several similar versions of ANSI/ISO standard
- All follow the same basic syntax and semantics

SQL statements can be used to:
- Create tables
- Insert and delete data in tables
- Create views
- Retrieve data with query statements

SQL Injection Attacks (SQLi)
- One of the most prevalent and dangerous network-based security threats
- Designed to exploit the nature of Web application pages
- Sends malicious SQL commands to the database server

DBMS
- DDL = data definition language
- DML = data manipulation language

Figure 5.1 DBMS Architecture

DML & DDL
- View - Limits what users can see

Exploiting transfer of knowledge across network
**Injection Technique**

The SQLi attack typically works by prematurely terminating a text string and appending a new command.

Because the inserted command may have additional strings appended to it before it is executed, the attacker terminates the injected string with a comment mark `--`.

Subsequent text is ignored at execution time.

---

**SQLi Attack Avenues**

- **User input**
  - Attackers input SQL commands by providing suitable crafted user input.

- **Server variables**
  - Attackers can forge the values that are placed in HTTP and network headers and exploit this vulnerability by placing data directly into the headers.

- **Second-order injection**
  - A malicious user could rely on data already present in the system or database to trigger an SQL injection attack, so when the attack occurs, the input that modifies the query to cause an attack does not come from the user, but from within the system itself.

- **Cookies**
  - An attacker could 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**
  - Applying user input that constructs an attack outside the realm of web requests.
Inferential Attack

- There is no actual transfer of data, but the attacker is able to reconstruct the information by sending particular requests and observing the resulting behavior of the Website/database server
- Include:
  - Illegal/logically incorrect queries
    - This attack lets an attacker gather important information about the type and structure of the backend database of a Web application
  - The attack is considered a preliminary, information-gathering step for other attacks
  - Blind SQL Injection
    - Allows attackers to infer the data present in a database system even when the system is sufficiently secure to not display any erroneous information back to the attacker

SQLi Countermeasures

- Three types:
  - Manual defensive coding practices
  - Parameterized query insertion
  - SQL DOM

- 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

Database Access Control

- Database access control system determines:
  - If the user has access to the entire database or just portions of it
  - What access rights the user has (create, insert, delete, update, read, write)

- Can support a range of administrative policies
  - Centralized administration
    - Small number of privileged users may grant and revoke access rights
  - Ownership-based administration
    - The creator of a table may grant and revoke access rights to the table
  - Decentralized administration
    - 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

Parameterized query insertion == Prepared Statement ≠ main defense

Most common
**SQL Access Controls**

- Two commands for managing access rights:
  - **Grant**
    - Used to grant one or more access rights or can be used to assign a user to a role
  - **Revoke**
    - Revokes the access rights

- Typical access rights are:
  - Select
  - Insert
  - Update
  - Delete
  - References

---

**Role-Based Access Control (RBAC)**

- Role-based access control eases administrative burden and improves security
- A database RBAC needs to provide the following capabilities:
  - Create and delete roles
  - Define permissions for a role
  - Assign and cancel assignment of users to roles
- Categories of database users:
  - **Application owner**
    - An end user who owns database objects as part of an application
  - **End user**
    - An end user who operates on database objects but does not own any of the database objects
  - **Administrator**
    - User who has administrative responsibility for part or all of the database

---

**Figure 5.6 Bob Revokes Privilege from David**
Inference Detection

- Some inference detection algorithm is needed for either of these approaches.
- Progress has been made in developing specific inference detection techniques for multilevel secure databases and statistical databases.
The database is typically the most valuable information resource for any organization. It is protected by multiple layers of security, including firewalls, authentication, general access control systems, DB access control systems, and database encryption. Encryption becomes the last line of defense in database security.

Encryption can be applied to the entire database, at the record level, the attribute level, or the individual field levels.

Disadvantages to encryption include:
- Key management: Authorized users must have access to the decryption key for the data for which they have access.
- Inflexibility: When part or all of the database is encrypted, it becomes more difficult to perform record searching.

Encryption causes performance degradation, and keys are a disadvantage.

**Summary**

- The need for database security
- Database management systems
- Relational databases
- SQL injection attacks
- Inference
- Database access control
  - SQL-based access definition
  - Cascading authorizations
  - Role-based access control
- Database encryption