Topic 9: Information Security and Privacy (Chapter 9 (Silberschatz et al.), Chapter 24 (Elmasri and Navathe), and Chapter 2 (Basse))

Modified for CS 4513

Additional References

- Ramez Elmasri and Shamkant Navathe, Chapter 24, “Database Security,” Fundamentals of Database Systems, 6th Edition, Pearson, 2011 (on the class website)

INFORMATION SECURITY

1 Introduction to Database Security Issues:

- Types of Security
  - Legal and ethical issues
  - Policy issues
  - System-related issues
  - The need to identify multiple security levels

1. Introduction to Database Security Issues (2)

- Threats to databases
  - Loss of **integrity**
  - Loss of **availability**
  - Loss of **confidentiality**

- Control measures: to protect databases against these types of threats, four kinds of countermeasures can be implemented:
  - **Access control**
  - **Inference control**
  - **Flow control**
  - **Data encryption**

1. Introduction to Database Security Issues (3)

- A DBMS typically includes a database security and authorization subsystem that is responsible for ensuring the security portions of a database against unauthorized access.
- Two types of database security mechanisms:
  - **Discretionary** security mechanisms
    - Used to grant privileges to users
  - **Mandatory** security mechanisms
    - Used to enforce multilevel security

1. Introduction to Database Security Issues (4)

- The security mechanism of a DBMS must include provisions for restricting access to the database as a whole
  - This function is called **access control** and is handled by creating user accounts and passwords to control the login process by the DBMS.

1. Introduction to Database Security Issues (5)

- The security problem associated with databases is that of controlling the access to a **statistical database**, which is used to provide statistical information or summaries of values based on various criteria.
  - The countermeasures to **statistical database security** problem is called **inference control measures.**

1. Introduction to Database Security Issues (6)

- Another security is that of **flow control**, which prevents information from flowing in such a way that it reaches unauthorized users.

- Channels that are pathways for information to flow implicitly in ways that violate the security policy of an organization are called **covert channels.**

1. Introduction to Database Security Issues (7)

- A final security issue is **data encryption**, which is used to protect sensitive data (such as credit card numbers) that is being transmitted via some type of communication network.
- The data is **encoded** using some **encoding algorithm**.
  - An unauthorized user who accesses encoded data will have difficulty deciphering it, but authorized users are given decoding or decrypting algorithms (or keys) to decipher data.

1.2 Database Security and the DBA

- The database administrator (**DBA**) is the central authority for managing a database system.
  - The DBA's responsibilities include
    - granting privileges to users who need to use the system
    - classifying users and data in accordance with the policy of the organization
- The DBA is responsible for the overall security of the database system.

1.2 Database Security and the DBA (2)

- The DBA has a DBA account in the DBMS
  - Sometimes these are called a system or super-user account
  - These accounts provide powerful capabilities such as:
    - 1. Account creation
    - 2. Privilege granting
    - 3. Privilege revocation
    - 4. Security level assignment
  - Action 1 is used to control access to the DBMS as a whole; 2 and 3 are used to control discretionary authorization; and 4 is used to control mandatory authorization.

1.3 Access Protection, User Accounts, and Database Audits

- Whenever a person or group of persons needs to access a database system, the individual or group must first apply for a user account.
  - The DBA will then create a new **account id** and **password** for the user if he/she deems there is a legitimate need to access the database.
- The user must log into the DBMS by entering account id and password whenever database access is needed.

1.3 Access Protection, User Accounts, and Database Audits (2)

- The database system must also **keep track of all operations** on the database that are applied by a certain user throughout **each login session**.
  - To keep a record of all updates applied to the database and of the particular user who applied each update, we can modify **system log**, which includes an entry for each operation applied to the database that may be required for recovery from a transaction failure or system crash.

1.3 Access Protection, User Accounts, and Database Audits (3)

- If any tampering with the database is suspected, a **database audit** is performed.
  - A database audit consists of reviewing the log to examine all accesses and operations applied to the database during a certain time period.
- A database log that is used mainly for security purposes is sometimes called an **audit trail**.

1.3 Access Protection, User Accounts, and Database Audits (4)

- Applications must log actions to an **audit trail** to detect who carried out an update or accessed some sensitive data.
- **Audit trails** are used after-the-fact to
  - detect security breaches
  - repair damage caused by security breach
  - trace who carried out the breach

2. Discretionary Access Control Based on Granting and Revoking Privileges

- The typical method of enforcing **discretionary access control** in a database system is based on the **granting** and **revoking privileges**.

2.1 Types of Discretionary Privileges

- The **account level**:
  - At this level, the DBA specifies the particular privileges that each account holds independently of the relations in the database.
- The **relation level** (or **table level**):
  - At this level, the DBA can control the privilege to access each individual relation or view in the database.

2.1 Types of Discretionary Privileges (2)

- The privileges at the **account level** apply to the capabilities provided to the account itself and can include
  - the **CREATE SCHEMA** or **CREATE TABLE** privilege to create a schema or base relation;
  - the **CREATE VIEW** privilege;
  - the **ALTER** privilege to apply schema changes such as adding or removing attributes from relations;
  - the **DROP** privilege to delete relations or views;
  - the **MODIFY** privilege to insert, delete, or update tuples;
  - and the **SELECT** privilege to retrieve information from the database by using a **SELECT** query.

2.1 Types of Discretionary Privileges (3)

- The second level of privileges applies to the **relation level**
  - This includes **base relations** and virtual (**view**) relations.
- The granting and revoking of privileges generally follow an authorization model for discretionary privileges known as the **access matrix model** where
  - The **rows** of a matrix M represents **subjects** (users, accounts, programs).
  - The **columns** represent **objects** (relations, records, columns, views, operations).
  - Each position **M(i,j)** in the matrix represents **the types of privileges (read, write, update)** that **subject i** holds on **object j**.

2.1 Types of Discretionary Privileges (4)

- To control the granting and revoking of relation privileges, each relation R in a database is assigned an **owner account**, which is typically the account that was used when the relation was created in the first place.
  - The owner of a relation is given <u>all</u> privileges on that relation.
  - The owner account holder can **pass privileges** on any of the owned relations to other users by **granting** privileges to their accounts.

2.1 Types of Discretionary Privileges (5)

- In SQL the following types of privileges can be granted on each individual relation R:
  - **SELECT** (retrieval or read) privilege on R:
    - Gives the account retrieval privilege.
    - In SQL this gives the account the privilege to use the **SELECT** statement to retrieve tuples from R.
  - **MODIFY** privileges on R:
    - This gives the account the capability to modify tuples of R.
    - In SQL this privilege is further divided into **UPDATE**, **DELETE**, and **INSERT** privileges to apply the corresponding SQL command to R.
    - In addition, both the **INSERT** and **UPDATE** privileges can specify that only certain attributes can be updated by the account.

2.1 Types of Discretionary Privileges (6)

- In SQL the following types of privileges can be granted on each individual relation R (cont.):
  - **REFERENCES** privilege on R:
    - This gives the account the capability to **reference** relation R when specifying integrity constraints.
    - The privilege can also be **restricted** to specific attributes of R.

- Notice that to create a **view**, the account must have **SELECT** privilege on all relations involved in the view definition.

2.2 Specifying Privileges Using Views

- The mechanism of **views** is an important discretionary authorization mechanism in its own right. For example,
  - If the owner A of a relation R wants another account B to be able to retrieve only some fields of R, then A can create a view V of R that includes only those attributes and then grant SELECT on V to B.
  - The same applies to limiting B to retrieving only certain tuples of R; a view V' can be created by defining the view by means of a query that selects only those tuples from R that A wants to allow B to access.

2.2 Specifying Privileges Using Views (2)

- In some cases, it is not desirable for all users to see the entire logical model (that is, all the actual relations stored in the database.)
- Consider a person who needs to know an instructor’s name and department, but not the salary. This person should see a relation described in SQL by

  select *ID*, *name*, *dept_name*
  from *instructor*

- A **view** provides a mechanism to hide certain data from the view of certain users.
- Any relation that is not of the conceptual model but is made visible to a user as a “virtual relation” is called a **view**.

2.2 Specifying Privileges Using Views (3)

- A view is defined using the **create view** statement which has the form

  create view *v* **as** < query expression >

  where <query expression> is any legal SQL expression. The view name is represented by *v.*
- Once a view is defined, the view name can be used to refer to the virtual relation that the view generates.
- View definition is not the same as creating a new relation by evaluating the query expression
  – Rather, a view definition causes the saving of an expression; the expression is substituted into queries using the view.

2.2 Specifying Privileges Using Views (4)

- Create a view of instructors without their salary
  create view *faculty* **as**
  select *ID*, *name*, *dept_name*
  from *instructor*
- Find all instructors in the Biology department
  select *name*
  from *faculty*
  where *dept_name* = 'Biology'
- Create a view of department salary totals
  create view *departments_total_salary* (*dept_name*, *total_salary*) **as**
  select *dept_name*, **sum** (*salary*)
  from *instructor*
  group by *dept_name*;

2.3 Revoking Privileges

- In some cases it is desirable to grant a privilege to a user temporarily. For example,
  - The owner of a relation may want to grant the **SELECT** privilege to a user for a specific task and then revoke that privilege once the task is completed.
  - Hence, a mechanism for **revoking** privileges is needed. In SQL, a **REVOKE** command is included for the purpose of **canceling privileges**.

2.4 Propagation of Privileges using the GRANT OPTION

- Whenever the owner A of a relation R grants a privilege on R to another account B, the privilege can be given to B with or without the **GRANT OPTION**.
- If the **GRANT OPTION** is given, this means that B can also grant that privilege on R to other accounts.
  - Suppose that B is given the **GRANT OPTION** by A and that B then grants the privilege on R to a third account C, also with **GRANT OPTION**. In this way, the privileges on R can **propagate** to other accounts without the knowledge of the owner of R.
  - If the owner account A now revokes the privilege granted to B, all the privileges that B propagated based on that privilege should automatically be revoked by the system.

2.5 An Example

- Suppose that the DBA creates four accounts
  - A1, A2, A3, A4
- and wants only A1 to be able to create base relations. Then the DBA must issue the following GRANT command in SQL:

```
GRANT CREATETAB TO A1;
```

2.5 An Example (2)

- User account A1 can create tables under the schema called **EXAMPLE**.
- Suppose that A1 **creates** the two base relations **EMPLOYEE** and **DEPARTMENT**
  - A1 is then **owner** of these two relations and hence all the relation privileges on each of them.
- Suppose that A1 wants to grant A2 the privilege to insert and delete tuples in both of these relations, but A1 does not want A2 to be able to propagate these privileges to additional accounts:

```
GRANT INSERT, DELETE ON
    EMPLOYEE, DEPARTMENT TO A2;
```

2.5 An Example (3)

EMPLOYEE

Name	Ssn	Bdate	Address	Sex	Salary	Dno

DEPARTMENT

Dnumber	Dname	Mgr_ssn

Figure 24.1
Schemas for the two relations EMPLOYEE and DEPARTMENT.

2.5 An Example (4)

- Suppose that A1 wants to allow A3 to retrieve information from either of the two tables and also to be able to propagate the SELECT privilege to other accounts.
- A1 can issue the command:

```
GRANT SELECT ON EMPLOYEE, DEPARTMENT
    TO A3 WITH GRANT OPTION;
```

- A3 can grant the **SELECT** privilege on the **EMPLOYEE** relation to A4 by issuing:

```
GRANT SELECT ON EMPLOYEE TO A4;
```

  – Notice that A4 can’t propagate the SELECT privilege because GRANT OPTION was not given to A4

2.5 An Example (5)

- Suppose that A1 decides to revoke the SELECT privilege on the EMPLOYEE relation from A3; A1 can issue:

```
REVOKE SELECT ON EMPLOYEE FROM A3;
```

- The DBMS must now automatically revoke the SELECT privilege on EMPLOYEE from A4, too, because A3 granted that privilege to A4 and A3 does not have the privilege any more.

2.5 An Example (6)

- Suppose that A1 wants to give back to A3 a limited capability to SELECT from the EMPLOYEE relation and wants to allow A3 to be able to propagate the privilege.
  - The limitation is to retrieve only the NAME, BDATE, and ADDRESS attributes and only for the tuples with DNO=5.
- A1 then creates the view:

```
CREATE VIEW A3EMPLOYEE AS
  SELECT NAME, BDATE, ADDRESS
  FROM EMPLOYEE
  WHERE DNO = 5;
```

- After the view is created, A1 can grant **SELECT** on the view A3EMPLOYEE to A3 as follows:

```
GRANT SELECT ON A3EMPLOYEE TO A3
   WITH GRANT OPTION;
```

2.5 An Example (7)

- Finally, suppose that A1 wants to allow A4 to update only the SALARY attribute of EMPLOYEE, then A1 can issue:

```
GRANT UPDATE ON EMPLOYEE (SALARY) TO
A4;
```

  - The **UPDATE** or **INSERT** privilege can specify particular attributes that may be updated or inserted in a relation.
  - Other privileges (**SELECT**, **DELETE**) are not attribute specific.

2.6 Role-Based Access Control

- Emerged rapidly in the 1990s for managing and enforcing security in large-scale enterprise-wide systems.
- Privileges are associated with organizational roles, rather than individual users; individual users are then assigned to appropriate roles.
- Roles can be created (or destroyed) and granted (or revoked) authorizations
  - **create role** instructor;
  - **grant** *instructor* **to Amit;**
- Privileges can be granted to roles
  - **grant select on** *takes* **to** *instructor*;
- Roles can be granted to users, as well as to other roles
  - **create role** *teaching_assistant;*
  - **grant** *teaching_assistant* **to** *instructor*;
    - *Instructor* inherits all privileges of *teaching_assistant*
- Chain of roles
  - **create role** *dean*;
  - **grant** *instructor* **to** *dean*;
  - **grant** *dean* **to** Satoshi;

3 Mandatory Access Control and Role-Based Access Control for Multilevel Security

- The discretionary access control techniques of granting and revoking privileges on relations have traditionally been the main security mechanism for relational database systems.
- This is an all-or-nothing method:
  - A user either has or does not have a certain privilege.
- In many applications, an **additional security policy** is needed that classifies data and users based on security classes.
  - This approach as **mandatory access control** would typically be **combined** with the discretionary access control mechanisms.

3 Mandatory Access Control and Role-Based Access Control for Multilevel Security (2)

- Typical **security classes** are Top Secret (TS), Secret (S), Confidential (C), and Unclassified (U), where TS is the highest level and U the lowest: TS ≥ S ≥ C ≥ U

- The commonly used model for multilevel security, known as the Bell-LaPadula model, classifies each **subject** (user, account, program) and **object** (relation, tuple, column, view, operation) into one of the security classifications, TS, S, C, or U:
  - **Clearance** (Classification) of a subject S as **class(S**) and **Classification** of an object O as **class(O)**.

3 Mandatory Access Control and Role-Based Access Control for Multilevel Security (3)

- Two restrictions are enforced on data access based on the subject/object classifications:
  - **Simple security property**: A subject S is not allowed read access to an object O unless class(S) ≥ class(O).
  - A subject S is not allowed to write an object O unless class(S) ≤ class(O). This is known as the **star property** (or * property).

3 Mandatory Access Control and Role-Based Access Control for Multilevel Security (4)

- To incorporate multilevel security notions into the relational database model, it is common to consider attribute values and tuples as data objects.
- Hence, each attribute A is associated with a **classification attribute C** in the schema, and each attribute value in a tuple is associated with a corresponding security classification.
- In addition, in some models, a **tuple classification** attribute TC is added to the relation attributes to provide a classification for each tuple as a whole.
- Hence, a **multilevel relation** schema R with n attributes would be represented as
  - $R(A_1,C_1,A_2,C_2, \ldots, A_n,C_n,TC)$
- where each Ci represents the classification attribute associated with attribute A_i.

4 SQL Injection

- One of the most common threats to a database system.
- Web programs and applications that access a database can send commands and data to the database, as well as display data retrieved from the database through the Web browser.
- In an SQL injection attack, the attacker injects a string input through the application, which changes or manipulates the SQL statements to the attacker’s advantage.

4 SQL Injection (2)

- Types of Injection Attacks:
  - SQL Manipulation:
    - is the most common type of injection attacks
    - changes an SQL command in the application, e.g. by adding a WHERE clause to a query, or expanding a query with additional query components using set operations (e.g. UNION, INTERSECTION, MINUS)
  - Code Injection:
    - Adds additional SQL statements to the existing SQL statements
  - Function Call Injection:
    - Inserts a database function or operating system function call into an SQL statement

4 SQL Injection (3)

- Suppose a Java program inputs a string *name* and constructs the query:
  - "select * from instructor where name = ’" + name + "’"
- Suppose the user, instead of entering a name, enters:
  - X’ or ’Y’ = ’Y
- then the resulting statement becomes:
  - "select * from instructor where name = ’" + "X’ or ’Y’ = ’Y" + "’"
  - which is:
    - select * from instructor where name = ’X’ or ’Y’ = ’Y’
    - the condition is always TRUE => the user can retrieve all instructor information

4 SQL Injection (4)

- An example of finding whether a web application is vulnerable to SQL injection attacks [http://www.packetsource.com]

- If the URL of a web page is:
  - http://www.prey.com/sample.jsp?param1=9
  - The SQL statement that the web application would use to retrieve the information from the database may look like this: SELECT column1, column2 FROM Table1 WHERE param1 = 9
  - After executing this query the database would return data in column1 and column2 for the rows which satisfy the condition param1 = 9. This data is processed by the server side code like servlets etc. and an HTML document is generated to display the information.

- To test the vulnerability of the web application, the attacker may modify the 'Where' clause by modifying the user inputs in the URL as follows:
  - http://www.prey.com/sample.jsp?param1=9 AND 1=1
  - if the database server executes the following query: SELECT column1, column2 FROM Table1 WHERE param1 = 9 AND 1=1 . If this query also returns the same information as before, then the application is susceptible to SQL injection

5 Flow Control

- **Flow control** regulates the distribution or flow of information among accessible objects.
- A **flow** between object X and object Y occurs when a program reads values from X and writes values into Y.
  - Flow controls check that information contained in some objects does not flow explicitly or implicitly into less protected objects.
- A **flow policy** specifies the channels along which information is allowed to move.
  - The simplest flow policy specifies just two classes of information:
    - confidential (C) and non-confidential (N)
  - and allows all flows except those from class C to class N.

5.1 Covert Channels

- A **covert channel** allows a transfer of information that violates the security or the policy.

- A **covert channel allows** information to pass from a higher classification level to a lower classification level through **improper means**.

6 Data Encryption

- **Encryption** is a means of maintaining secure data in an insecure environment.
- Data may be ***encrypted*** when database authorization provisions do not offer sufficient protection.
- **Encryption** consists of applying an **encryption algorithm** to transform the data into another form that is unreadable using some pre-specified **encryption key.**
- The resulting data has to be **decrypted** using a **decryption key** to recover the original data.
- Properties of a good encryption technique:
  - Relatively simple for authorized users to encrypt and decrypt data.
  - Encryption scheme depends not only on the secrecy of the algorithm but also on the secrecy of a parameter of the algorithm called the encryption key.
  - Extremely difficult for an intruder to determine the encryption key.

INFORMATION PRIVACY

1. Privacy and Computer Technology

- Three aspects of privacy:
  - Freedom from intrusion – being left alone
  - Control of information about oneself
  - Freedom from surveillance (from being followed, tracked, watched, and eavesdropped on)

1. Privacy and Computer Technology (2)

- Privacy threats come in several categories:
  - Intentional, institutional uses of personal information
  - Unauthorized use or release by “insiders”
  - Theft of information
  - Inadvertent leakage of information through negligence or carelessness
  - Our own actions

1. Privacy and Computer Technology (3)

- New technologies, new risks:
  - Computers, the Internet, mobile devices, etc.
  - Increases in speed, storage space, and connectivity make the collection, searching, analysis, storage, access, and distribution of huge amounts of information much easier, cheaper and faster than ever before.
  - Thousands of databases, both government and private, contain people's personal information.
  - Great benefits, but also great privacy threats.

1. Privacy and Computer Technology (4)

- People often are not aware that information about them and their activities are being collected and saved.
- Once data goes on the Internet or into a database, it seems to last forever.
- We cannot directly protect information about ourselves. We must depend on the businesses and organizations that hold it to protect it from thieves and accidental leaks and government prying.

1. Privacy and Computer Technology (5)

- Principles for data collection and use:
  - Inform people when personally identifiable information about them is collected, what is collected, and how it will be used.
  - Collect only the data needed.
  - Offer a way for people to opt out from mailing lists, advertising, transfer of their data to other parties, and other secondary uses.
  - Keep data only as long as needed.
  - Maintain accuracy of data. Where appropriate and reasonable, provide a way for people to access and correct data stored about them.
  - Protect security of data (from theft and from accidental leaks). Provide stronger protection for sensitive data.
  - Develop policies for responding to law enforcement requests for data.

1. Privacy and Computer Technology (6)

- Need to consider privacy issues early to include privacy protection features in the technology (e.g. database) design and implementation process.

2. Diverse Privacy Topics

- 2.1. Marketing, personalization and consumer dossiers
  - Targeted and personalized marketing using consumers' information collected
  - Marketing by businesses, political parties and organizations to:
    - find new customers, members, or voters and encourage old ones to continue
    - advertise products, services, or causes
    - determine how to price products and when and to whom to offer discounts
  - Consumers may not be aware of all the information collected about them => potential privacy threats

2. Diverse Privacy Topics (2)

- 2.2. Location Tracking
  - Devices such as GPS, cell phones, and smart phones enable development of many location-based applications.
  - The added detailed information about our current location and past movements to the pool of information already available about us => potential privacy threats.

2. Diverse Privacy Topics (3)

- 2.3. Stolen and Lost Data
  - Criminals steal personal data by
    - hacking into computer systems
    - stealing computers and disks
    - buying or requesting records under false pretenses
    - bribing employees of companies that store the data
  - Legal and Ethical Responsibility of those who maintain the data:
    - must anticipate risks and prepare for them to protect the data => must implement **strong security measures**
  - Federal law provides penalties for improper disclosure of personal data:
    - Health Insurance Portability and Accountability Act HIPPA: penalties for medical organizations
    - Privacy Act: penalties for government agencies

2. Diverse Privacy Topics (4)

- 2.4. Privacy-Preserving Data Mining and Data Integration
  - Must protect privacy while mining data for new patterns/knowledge
    - The resulting new knowledge must not reveal private information
  - Must protect privacy while integrating data from multi sources
    - The resulting integrated data must not reveal private information

Relationship between Information Security and Information Privacy

1. Relationship between Information Security and Information Privacy

- Security in information technology refers to many aspects of protecting a system from unauthorized use, including
  - Authentication of users
  - Information encryption
  - Access control
  - Firewall policies
  - Intrusion detection
- Privacy examines how well the use of personal information that the system acquires about a user conforms to the explicit or implicit assumptions regarding that use.
- From an end user perspective, privacy can be considered from two different perspectives: preventing storage of personal information versus ensuring appropriate use of personal information.

1. Relationship between Information Security and Information Privacy (2)

- In summary:
  - Privacy is the ability of individuals to control the terms under which their personal information is acquired and used.
  - Security involves technology to ensure that information is appropriately protected.
  - Security is a building block for privacy to exist.
  - Privacy involves mechanisms to support compliance with some basic principles and other explicitly stated policies.

END OF TOPIC 9