**Module 3**

**Data Security**

**Motivation:**

In the last few years, the amount of information stored in electronic media has increased significantly. Information which is stored in some medium is usually called data. As data is being used extensively nowadays there are many users and some of them become malicious and try to pose threat to the data. Hence there is a need to protect/secure data from malicious users and unauthorized access.

**Syllabus:**

|  |  |  |  |
| --- | --- | --- | --- |
| Lecture no | Content | Duration  (Hr) | Self-Study (Hrs) |
| 1 | Introduction to Database Security Issues; authorization | 1 | 1 |
| 2 | Discretionary Access Control Based on Granting and Revoking Privileges | 1 | 2 |
| 3 | Mandatory Access Control and Role-Based | 1 | 2 |
| 4 | Access Control for Multilevel Security - SQL Injection | 1 | 2 |
| 5 | Introduction to Statistical Database Security | 1 | 2 |
| 6 | Introduction to Flow Control | 1 | 2 |

**Learning Objectives:**

Learners shall be able to:

1. To list the different database securities issues and explain

2. To understand the concept behind different access control mechanisms

3. To explain the concept of statistical database security

4. To explain about flow control

**Theoretical Background:**

Database security encompasses a range of security controls designed to protect the Database Management System (DBMS). The types of database security measures your business should use include protecting the underlying infrastructure that houses the database such as the network and servers), securely configuring the DBMS, and the access to the data itself.

Safeguarding the data any company collects and manages is of utmost importance. Database security can guard against a compromise of the database, which can lead to financial loss, reputation damage, consumer confidence disintegration, brand erosion, and non-compliance of government and industry regulation.

Database security safeguards defend against a myriad of security threats and can help protect your enterprise from:

* Deployment failure
* Excessive privileges
* Privilege abuse
* Platform vulnerabilities
* Unmanaged sensitive data
* Backup data exposure
* Weak authentication
* Database injection attacks

**Key Definitions:**

**Database Security:** Database security refers to the range of tools, controls, and measures designed to establish and preserve database confidentiality, integrity, and availability.

**Threat:** Any situation or event, whether intentionally or incidentally, can cause damage, which can reflect an adverse effect on the database structure and, consequently, the organization. A threat may occur by a situation or event involving a person or the action or situations that are probably to bring harm to an organization and its database.

**Authorization:** Authorization is the process where the database manager gets information about the authenticated user. Part of that information is determining which database operations the user can perform and which data objects a user can access.

**Access Control**: Database access control is a method of allowing access to company's sensitive data only to those people (database users) who are allowed to access such data and to restrict access to unauthorized persons. It includes two main components: authentication and authorization.

**Authentication:** Database authentication is the process or act of confirming that a user who is attempting to log in to a database is authorized to do so, and is only accorded the rights to perform activities that he or she has been authorized to do.

**Privacy:** Data privacy or information privacy is a branch of data security concerned with the proper handling of data – consent, notice, and regulatory obligations. More specifically, practical data privacy concerns often revolve around: Whether or how data is shared with third parties.

**Data Integrity:** Data integrity is a fundamental component of information security. In its broadest use, “data integrity” refers to the accuracy and consistency of data stored in a database.

**Discretionary Access Control:** Discretionary access control (DAC) is a type of access control defined by the Trusted Computer System Evaluation Criteria "as a means of restricting access to objects based on the identity of subjects and/or groups to which they belong.

**Mandatory Access Control:** Mandatory access control (MAC) refers to a type of access control by which the operating system constrains the ability of a subject or initiator to access or generally perform some sort of operation on an object or target.

**Role-based Access Control:** Role-based access control (RBAC) is a method of restricting network access based on the roles of individual users within an enterprise. RBAC lets employees have access rights only to the information they need to do their jobs and prevents them from accessing information that doesn't pertain to them.

**SQL Injection:** SQL injection, also known as SQLI, is a common attack vector that uses malicious SQL code for backend database manipulation to access information that was not intended to be displayed. This information may include any number of items, including sensitive company data, user lists or private customer details.

**Statistical Database Security:** Statistical database security focuses on the protection of confidential individual values stored in so called statistical databases and used for statistical purposes. The goal of statistical data protection is to maximize the privacy while minimizing the information loss.

**Flow Control Security:** Flow control prevents data from being transferred in such a way that it can be accessed by unauthorized agents. A flow policy lists out the channels through which information can flow. It also defines security classes for data as well as transactions.

**Course Content:**

**Lecture 1**

**Introduction to Database Security Issues; authorization:**

Database security is the technique that protects and secures the database against intentional or accidental threats. Security concerns will be relevant not only to the data resides in an organization's database: the breaking of security may harm other parts of the system, which may ultimately affect the database structure. Consequently, database security includes hardware parts, software parts, human resources, and data. To efficiently do the uses of security needs appropriate controls, which are distinct in a specific mission and purpose for the system. The requirement for getting proper security while often having been neglected or overlooked in the past days; is now more and more thoroughly checked by the different organizations.

**Types of Security:**

Database security is a broad area that addresses many issues, including the following:

* Various **legal and ethical issues** regarding the right to access certain information—for example, some information may be deemed to be **private** and cannot be accessed legally by unauthorized organizations or persons. In the United States, there are numerous laws governing **privacy of information.**
* **Policy issues** at the governmental, institutional, or corporate level as to what kinds of information should not be made publicly available—for example, **credit ratings** and **personal medical records.**
* **System-related issues** such as the system levels at which various security functions should be enforced—for example, whether a security function should be handled at the physical **hardware level,** the **operating system level,** or the **DBMS level.**

The need in some organizations is to identify **multiple security levels** and to categorize the data and

users based on these classifications—for example, **top secret, secret, confidential,** and **unclassified.** The security policy of the organization with respect to permitting access to various classifications of data must be enforced.

**Threats to Databases and Protection measures:** Threats to databases can result in the loss or degradation of some or all of the following commonly accepted security goals: *integrity, availability,* and *confidentiality* (protection from unauthorized disclosure). To protect databases against these types of threats, it is common to implement four kinds of control measures: (1) access control, (2) inference control, (3) flow control, and (4) encryption.

Control Measures:

Four main control measures are used to provide security of data in databases:

* **Access control:** creating user accounts and passwords to control the login process by the DBMS.
* **Inference control:** Cannot deduce or infer certain facts concerning individuals from the DB.
* **Flow control:** Protect information from reaching unauthorized users.
* **Data encryption:** The data is encoded using some coding algorithm.

A DBMS typically includes a database security and authorization subsystem that is responsible for ensuring the security of portions of a database against unauthorized access. It is now customary to refer to two types of database security mechanisms:

* **Discretionary security mechanisms:** These are used to grant privileges to users, including the capability to access specific data files, records, or fields in a specified mode (such as read, insert, delete, or update).
* **Mandatory security mechanisms:** These are used to enforce multilevel security by classifying the data and users into various security classes (or levels) and then implementing the appropriate security policy of the organization. For example, a typical security policy is to permit users at a certain classification (or clearance) level to see only the data items classified at the user's own (or lower) classification level. An extension of this is role-based security, which enforces policies and privileges based on the concept of organizational roles.

**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 and classifying users and data in accordance with the policy of the organization. The DBA has a DBA account in the DBMS, sometimes called a system or superuser account, which provides powerful capabilities that are not made available to regular database accounts and users.2 DBA-privileged commands include commands for granting and revoking privileges to individual accounts, users, or user groups and for performing the following types of actions:

1. Account creation. This action creates a new account and password for a user or a group of users to enable access to the DBMS.

2. Privilege granting. This action permits the DBA to grant certain privileges to certain accounts.

3. Privilege revocation. This action permits the DBA to revoke (cancel) certain privileges that were previously given to certain accounts.

4. Security level assignment. This action consists of assigning user accounts to the appropriate security clearance level.

The DBA is responsible for the overall security of the database system. Action 1 in the preceding list is used to control access to the DBMS as a whole, whereas actions 2 and 3 are used to control discretionary database authorization, and action 4 is used to control mandatory authorization.

*Let’s check the take away from this lecture*

1) Prevention of access to the database by unauthorized used is referred to as: \*

a. Integrity

b. Productivity

**c. Security**

d. Reliability

2) In role-based access control, each user is assigned one or more roles, and the roles determine

which parts of the system the user is allowed to access.

**a. True**

b. False

3) Data integrity means

a. providing first access to stored data

**b. ensuring correctness and consistency and correction of data**

c. providing data sharing

d. none of the above

Exercise

Q.1 Define the term database security.

Q.2 List the different access control mechanisms.

Q.3 What is a threat?

**Learning from this lecture**: Learners will be able to understand the need for database security and authorization.

**Lecture 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 of **privileges**. Let us consider privileges in the context of a relational DBMS. In particular, we will discuss a system of privileges somewhat similar to the one originally developed for the SQL language. Many current relational DBMSs use some variation of this technique. The main idea is to include statements in the query language that allow the DBA and selected users to grant and revoke privileges.

**1. Types of Discretionary Privileges**

In SQL2 and later versions, the concept of an **authorization identifier** is used to refer, roughly speaking, to a user account (or group of user accounts). For simplicity, we will use the words *user* or *account* interchangeably in place of *authorization* *identifier.*The DBMS must provide selective access to each relation in the databasebased on specific accounts. Operations may also be controlled; thus, having an account does not necessarily entitle the account holder to all the functionality provided by the DBMS. Informally, there are two levels for assigning privileges to use the database system:

**The account level:**At this level, the DBA specifies the particular privilegesthat each account holds independently of the relations in the database.

**The relation (or table) level:**At this level, the DBA can control the privilegeto access each individual relation or view in the database.

**References privilege on *R:***This gives the account the capability to*reference*(or refer to) a relation *R* when specifying integrity constraints. This privilege can also be restricted to specific attributes of *R*.

Note that to create a view; the account must have the SELECT privilege on *all relations*involved in the view definition in order to specify the query that correspondsto the view.

**2. Specifying Privileges through the Use of 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. We will illustrate this discussion with the example.

**3. Revoking of 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.

**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 the GRANT OPTION. In this way, 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.

 It is possible for a user to receive a certain privilege from two or more sources. For example, A4 may receive a certain UPDATE *R* privilege from *both* A2 and A3. In such a case, if A2 revokes this privilege from A4, A4 will still continue to have the privilege by virtue of having been granted it from A3. If A3 later revokes the privilege from A4, A4 totally loses the privilege. Hence, a DBMS that allows propagation of privileges must keep track of how all the privileges were granted so that revoking of priv-ileges can be done correctly and completely.

**5. An Example to Illustrate Granting and Revoking of Privileges**

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

GRANT CREATETAB TO A1;

The CREATETAB (create table) privilege gives account A1 the capability to create new database tables (base relations) and is hence an *account privilege.* This privilege was part of earlier versions of SQL but is now left to each individual system implementation to define.

 In SQL2 the same effect can be accomplished by having the DBA issue a CREATE SCHEMA command, as follows:

CREATE SCHEMA EXAMPLE AUTHORIZATION A1;

User account A1 can now create tables under the schema called EXAMPLE. To con-tinue our example, suppose that A1 creates the two base relations EMPLOYEE and DEPARTMENT shown in Figure 24.1; A1 is then the **owner** of these two relations and hence has *all the relation privileges* on each of them.

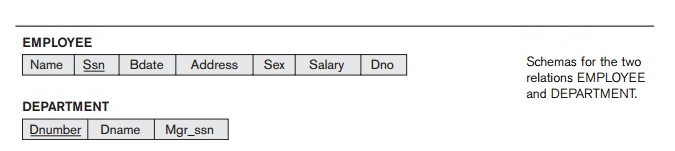
Next, suppose that account A1 wants to grant to account A2 the privilege to insert and delete tuples in both of these relations. However, A1 does not want A2 to be able to propagate these privileges to additional accounts. A1 can issue the following com-mand:

GRANT INSERT, DELETE ON EMPLOYEE, DEPARTMENT TO A2;

Notice that the owner account A1 of a relation automatically has the GRANT OPTION, allowing it to grant privileges on the relation to other accounts. However, account A2 cannot grant INSERT and DELETE privileges on the EMPLOYEE and DEPARTMENT tables because A2 was not given the GRANT OPTION in the preceding command.

Next, suppose that A1 wants to allow account 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 following command:

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



The clause WITH GRANT OPTION means that A3 can now propagate the privilege to other accounts by using GRANT. For example, A3 can grant the SELECT privilege on the EMPLOYEE relation to A4 by issuing the following command:

GRANT SELECT ON EMPLOYEE TO A4;

Notice that A4 cannot propagate the SELECT privilege to other accounts because the GRANT OPTION was not given to A4.

Now suppose that A1 decides to revoke the SELECT privilege on the EMPLOYEE relation from A3; A1 then can issue this command:

REVOKE SELECT ON EMPLOYEE FROM A3;

The DBMS must now revoke the SELECT privilege on EMPLOYEE from A3, and it must also *automatically revoke* the SELECT privilege on EMPLOYEE from A4. This is because A3 granted that privilege to A4, but A3 does not have the privilege any more.

Next, 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 can create the following 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;

Finally, suppose that A1 wants to allow A4 to update only the Salary attribute of EMPLOYEE; A1 can then issue the following command:

GRANT UPDATE ON EMPLOYEE (Salary) TO A4;

The UPDATE and INSERT privileges can specify particular attributes that may be updated or inserted in a relation. Other privileges (SELECT, DELETE) are not attrib-ute specific, because this specificity can easily be controlled by creating the appro-priate views that include only the desired attributes and granting the corresponding privileges on the views. However, because updating views is not always possible (see Chapter 5), the UPDATE and INSERT privileges are given the option to specify the particular attributes of a base relation that may be updated.

**6. Specifying Limits on Propagation of Privileges**

 Techniques to limit the propagation of privileges have been developed, although they have not yet been implemented in most DBMSs and *are not a part* of SQL. Limiting **horizontal propagation** to an integer number *i* means that an account *B* given the GRANT OPTION can grant the privilege to at most *i* other accounts.

**Vertical propagation**is more complicated; it limits the depth of the granting ofprivileges. Granting a privilege with a vertical propagation of zero is equivalent to granting the privilege with *no* GRANT OPTION. If account *A* grants a privilege to account *B* with the vertical propagation set to an integer number *j* > 0, this means that the account *B* has the GRANT OPTION on that privilege, but *B* can grant the privilege to other accounts only with a vertical propagation *less than j.* In effect, vertical propagation limits the sequence of GRANT OPTIONS that can be given from one account to the next based on a single original grant of the privilege.

We briefly illustrate horizontal and vertical propagation limits—which are *not* *available*currently in SQL or other relational systems—with an example. Supposethat A1 grants SELECT to A2 on the EMPLOYEE relation with horizontal propagation equal to 1 and vertical propagation equal to 2. A2 can then grant SELECT to at most one account because the horizontal propagation limitation is set to 1. Additionally, A2 cannot grant the privilege to another account except with vertical propagation set to 0 (no GRANT OPTION) or 1; this is because A2 must reduce the vertical propagation by at least 1 when passing the privilege to others. In addition, the horizontal propagation must be less than or equal to the originally granted horizontal propagation. For example, if account *A* grants a privilege to account *B* with the horizontal propagation set to an integer number *j* > 0, this means that *B* can grant the privilege to other accounts only with a horizontal propagation *less than or* *equal to j.*As this example shows, horizontal and vertical propagation techniques aredesigned to limit the depth and breadth of propagation of privileges.

*Let’s check the take away from this lecture*

1) The database administrator who authorizes all the new users, modifies the database and takes grants privilege is

a) Super user

b) Administrator

c) Operator of operating system

**d) All of the mentioned**

2) Which of the following is used to provide privilege to only a particular attribute?

a) Grant select on employee to Ajay

**b) Grant update (budget) on department to Rahul**

c) Grant update (budget, salary, Rate) on department to Rahul

d) Grant delete to Ajay

3) Which of the following statement is used to remove the privilege from the user Ajay?

a) Remove update on department from Ajay

**b) Revoke update on employee from Ajay**

c) Delete select on department from Rahul

d) Grant update on employee from Ajay

Exercise

Q.1 Define Discretionary Access Control.

Q.2 Illustrate with an Example for Granting and Revoking of Privileges.

Q.3 Explain vertical and horizontal propagation.

**Learning from this lecture**: Learners will be able to understand the concept of Discretionary Access Control and can give examples on that.

**Lecture 3**

**Mandatory Access Control and Role-Based Access Control**

**Mandatory Access Control**

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 where a user either has or does not have a certain privilege. In many applications, and additional security policy is needed that classifies data and users based on security classes. This approach known as mandatory access control would typically be combined with the discretionary access control mechanisms.

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, T, S, C, or U: The clearance (classification) of a subject S is referred to as class(S) and the classification of an object O as class(O). 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 known as the **star property** (or \* property).

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 I s 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(A1,C1,A2,C2, …, An,Cn,TC)

where each Ci represents the classification attribute associated with attribute Ai.

The value of the **TC** attribute in each tuple t – which is the highest of all attribute classification values within t – provides a general classification for the tuple itself, whereas each Ci provides a finer security classification for each attribute value within the tuple.

The apparent key of a multilevel relation is the set of attributes that would have formed the primary key in a regular (single-level) relation. A multilevel relation will appear to contain different data to subjects (users) with different clearance levels. In some cases, it is possible to store a single tuple in the relation at a higher classification level and produce the corresponding tuples at a lower-level classification through a process known as **filtering**. In other cases, it is necessary to store two or more tuples at different classification levels with the same value for the **apparent key**. This leads to the concept of **polyinstantiation** where several tuples can have the same apparent key value but have different attribute values for users at different classification levels.

In general, the **entity integrity** rule for multilevel relations states that all attributes that are members of the apparent key must not be null and must have the same security classification within each individual tuple. In addition, all other attribute values in the tuple must have a security classification greater than or equal to that of the apparent key. This **constraint** ensures that a user can see the key if the user is permitted to see any part of the tuple at all.

Other integrity rules, called **null integrity** and **inter instance integrity**, informally ensure that if a tuple value at some security level can be filtered (derived) from a higher-classified tuple, then it is sufficient to store the higher-classified tuple in the multilevel relation.

**Comparing Discretionary Access Control and Mandatory Access Control**

Discretionary Access Control (DAC) policies are characterized by a high degree of flexibility, which makes them suitable for a large variety of application domains. The main drawback of DAC models is their vulnerability to malicious attacks, such as Trojan horses embedded in application programs.

By contrast, mandatory policies ensure a high degree of protection in a way, they prevent any illegal flow of information. Mandatory policies have the drawback of being too rigid and they are only applicable in limited environments. In many practical situations, discretionary policies are preferred because they offer a better trade-off between security and applicability.

**Role-Based Access Control**

**Role-based access control (RBAC)** emerged rapidly in the 1990s as a proven technology for managing and enforcing security in large-scale enterprise wide systems. Its basic notion is that permissions are associated with roles, and users are assigned to appropriate roles. Roles can be created using the **CREATE ROLE** and **DESTROY ROLE** commands. The **GRANT** and **REVOKE** commands discussed under DAC can then be used to assign and revoke privileges from roles.

RBAC appears to be a viable alternative to traditional discretionary and mandatory access controls; it ensures that only authorized users are given access to certain data or resources. Many DBMSs have allowed the concept of roles, where privileges can be assigned to roles. Role hierarchy in RBAC is a natural way of organizing roles to reflect the organization’s lines of authority and responsibility.

Another important consideration in RBAC systems is the possible temporal constraints that may exist on roles, such as time and duration of role activations, and timed triggering of a role by an activation of another role. Using an RBAC model is highly desirable goal for addressing the key security requirements of Web-based applications. In contrast, discretionary access control (DAC) and mandatory access control (MAC) models lack capabilities needed to support the security requirements emerging enterprises and Web-based applications.

*Let’s check the take away from this lecture*

1) Which of the following is NOT a valid access control mechanism?

a) DAC (Discretionary Access Control) list.

**b) SAC (Subjective Access Control) list.**

c) MAC (Mandatory Access Control) list.

d) RBAC (Role Based Access Control) list.

2) Which of the following best describes an access control mechanism in which access control decisions are based on the responsibilities that an individual user or process has in an organization?

a) MAC (Mandatory Access Control)

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

c) DAC (Discretionary Access Control)

d) None of the above.

3) Which of the following access control methods relies on user security clearance and data classification?

a) RBAC (Role Based Access Control).

b) NDAC (Non-Discretionary Access Control).

**c) MAC (Mandatory Access Control).**

d) DAC (Discretionary Access Control).

Exercise

Q.1 Define Role-Based Access Control.

Q.2 List and define subject/object classifications in Mandatory Access Control.

Q.3 Compare Discretionary access control and Mandatory access control.

**Learning from this lecture**: Learners will be able to understand the concept of Mandatory Access Control, Role-Based Access Control and can give examples on that.

**Lecture 4**

**Access Control for Multilevel Security - SQL Injection**

SQL injection is one of the most common threats to a database system. We will discuss it in detail later in this section. Some of the other frequent attacks on databases are:

* Unauthorized privilege escalation: This attack is characterized by an individual attempting to elevate his or her privilege by attacking vulnerable points in the database systems.
* Privilege abuse: Whereas unauthorized privilege escalation is done by an unauthorized user, this attack is performed by a privileged user. For example, an administrator who is allowed to change student information can use this privilege to update student grades without the instructor’s permission.
* Denial of service: A denial of service (DOS) attack is an attempt to make resources unavailable to its intended users. It is a general attack category in which access to network applications or data is denied to intended users by overflowing the buffer or consuming resources.
* Weak authentication: If the user authentication scheme is weak, an attacker can impersonate the identity of a legitimate user by obtaining her login credentials.

**SQL Injection Methods**

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 statement to the attacker’s advantage. An SQL injection attack can harm the database in various ways, such as unauthorized manipulation of the database or retrieval of sensitive data. It can also be used to execute system-level commands that may cause the system to deny service to the application. This section describes types of injection attacks like code injection and function call injection.

**SQL Manipulation:**

A manipulation attack, which is the most common type of injection attack, changes an SQL command in the application—for example, by adding conditions to the WHERE-clause of a query, or by expanding a query with additional query components using set operations such as UNION, INTERSECT, or MINUS. Other types of manipulation attacks are also possible. A typical manipulation attack occurs during database login. For example, suppose that a simplistic authentication procedure issues the following query and checks to see if any rows were returned:

SELECT \* FROM users WHERE username = ‘jake’ and PASSWORD = ‘jakespasswd’ ;

The attacker can try to change (or manipulate) the SQL statement by changing it as follows:

SELECT \* FROM users WHERE username = ‘jake’ and (PASSWORD = ‘jakespasswd’ or ‘x’ = ‘x’);

As a result, the attacker who knows that ‘jake’ is a valid login of some user is able to log into the database system as ‘jake’ without knowing his password and is able to do everything that ‘jake’ may be authorized to do to the database system.

1. **Code Injection:**

This type of attack attempts to add additional SQL statements or commands to the existing SQL statement by exploiting a computer bug, which is caused by processing invalid data. The attacker can inject or introduce code into a computer program to change the course of execution. Code injection is a popular technique for system hacking or cracking to gain information.

1. **Function Call Injection:**

In this kind of attack, a database function or operating system function call is inserted into a vulnerable SQL statement to manipulate the data or make a privileged system call. For example, it is possible to exploit a function that performs some aspect related to network communication. In addition, functions that are contained in a customized database package, or any custom database function, can be executed as part of an SQL query. In particular, dynamically created SQL queries can be exploited since they are constructed at runtime. For example, the dual table is used in the FROM clause of SQL in Oracle when a user needs to run SQL that does not logically have a table name. To get today’s date, we can use: SELECT SYSDATE FROM dual; The following example demonstrates that even the simplest SQL statements can be vulnerable.

SELECT TRANSLATE (‘user input’, ‘from\_string’, ‘to\_string’) FROM dual;

Here, TRANSLATE is used to replace a string of characters with another string of characters. The TRANSLATE function above will replace the characters of the ‘from\_string’ with the characters in the ‘to\_string’ one by one. This means that the f will be replaced with the t, the r with the o, the o with the \_, and so on. This type of SQL statement can be subjected to a function injection attack. Consider the following example:

SELECT TRANSLATE (“ || UTL\_HTTP.REQUEST (‘http://129.107.2.1/’) || ”, ‘98765432’, ‘9876’) FROM dual;

The user can input the string (“ || UTL\_HTTP.REQUEST (‘http://129.107.2.1/’) ||”), where || is the concatenate operator, thus requesting a page from a Web server. UTL\_HTTP makes Hypertext Transfer Protocol (HTTP) callouts from SQL. The REQUEST object takes a URL (‘http://129.107.2.1/’ in this example) as a parameter, contacts that site, and returns the data (typically HTML) obtained from that site. The attacker could manipulate the string he inputs, as well as the URL, to include other functions and do other illegal operations. Here a dummy example is used to show conversion of ‘98765432’ to ‘9876’, but the user’s intent would be to access the URL and get sensitive information. The attacker can then retrieve useful information from the database server—located at the URL that is passed as a parameter—and send it to the Web server (that calls the TRANSLATE function).

**Risks Associated with SQL Injection**

SQL injection is harmful and the risks associated with it provide motivation for attackers. Some of the risks associated with SQL injection attacks are explained below.

* **Database fingerprinting:** The attacker can determine the type of database being used in the backend so that he can use database-specific attacks that correspond to weaknesses in a particular DBMS.
* **Denial of service:** The attacker can flood the server with requests, thus denying service to valid users, or the attacker can delete some data.
* **Bypassing authentication:** This is one of the most common risks, in which the attacker can gain access to the database as an authorized user and perform all the desired tasks.
* **Identifying injectable parameters:** In this type of attack, the attacker gathers important information about the type and structure of the back-end database of a Web application. This attack is made possible by the fact that the default error page returned by application servers is often overly descriptive.
* **Executing remote commands:** This provides attackers with a tool to execute arbitrary commands on the database. For example, a remote user can execute stored database procedures and functions from a remote SQL interactive interface.
* **Performing privilege escalation:** This type of attack takes advantage of logical flaws within the database to upgrade the access level.

**Protection Techniques against SQL Injection**

Protection against SQL injection attacks can be achieved by applying certain programming rules to all Web-accessible procedures and functions. This section describes some of these techniques. Bind Variables (Using Parameterized Statements). The use of bind variables protects against injection attacks and also improves performance. Consider the following example using Java and JDBC:

PreparedStatement stmt = conn.prepareStatement (“SELECT \* FROM EMPLOYEE WHERE EMPLOYEE\_ID=? AND PASSWORD=?”);

stmt.setString(1, employee\_id); stmt.setString(2, password);

Instead of embedding the user input into the statement, the input should be bound to a parameter. In this example, the input ‘1’ is assigned (bound) to a bind variable 1146 Chapter 30 Database Security ‘employee\_id’ and input ‘2’ to the bind variable ‘password’ instead of directly passing string parameters.

**Filtering Input (Input Validation):** This technique can be used to remove escape characters from input strings by using the SQL Replace function. For example, the delimiter single quote (‘) can be replaced by two single quotes (‘’). Some SQL manipulation attacks can be prevented by using this technique, since escape characters can be used to inject manipulation attacks. However, because there can be a large number of escape characters, this technique is not reliable.

**Function Security:** Database functions, both standard and custom, should be restricted, as they can be exploited in the SQL function injection attacks.

*Let’s check the take away from this lecture*

1) SQL injection is an attack in which \_\_\_\_\_\_\_\_\_ code is inserted into strings that are later passed to an instance of SQL Server.

**a) malicious**

b) redundant

c) clean

d) non malicious

2) Point out the correct statement.

a) Parameterized data cannot be manipulated by a skilled and determined attacker

**b) Procedure that constructs SQL statements should be reviewed for injection vulnerabilities**

c) The primary form of SQL injection consists of indirect insertion of code

d) None of the mentioned

3) Attackers follow a certain methodology to perform SQL-injection attacks to ensure that these attacks are successful by analyzing all the possible methods to perform the attack

**a) True**

b) False

Exercise

Q.1 List out the risks associated with SQL Injection.

Q.2 Explain the protection techniques against SQL Injection.

Q.3 Define SQL injection attack.

**Learning from this lecture**: Learners will be able to understand the concept of multi level security and SQL injection attack.

**Lecture 5**

**Introduction to Statistical Database Security**

Statistical databases are used mainly to produce statistics on various populations. The database may contain confidential data on individuals, which should be protected from user access. Users are permitted to retrieve statistical information on the populations, such as averages, sums, counts, maximums, minimums, and standard deviations.

A population is a set of tuples of a relation (table) that satisfy some selection condition. Statistical queries involve applying statistical functions to a population of tuples. For example, we may want to retrieve the *number* of individuals in a population or the *average income* in the population. However, statistical users are not allowed to retrieve individual data, such as the income of a specific person. Statistical database security techniques must prohibit the retrieval of individual data. This can be achieved by prohibiting queries that retrieve attribute values and by allowing only queries that involve statistical aggregate functions such as COUNT, SUM, MIN, MAX, AVERAGE, and STANDARD DEVIATION. Such queries are sometimes called statistical queries. It is DBMS’s responsibility to ensure confidentiality of information about individuals, while still providing useful statistical summaries of data about those individuals to users. Provision of privacy protection of users in a statistical database is paramount.

In some cases it is possible to infer the values of individual tuples from a sequence statistical queries. This is particularly true when the conditions result in a population consisting of a small number of tuples.

*Let’s check the take away from this lecture*

1) Lack of access control policy is a \_\_\_\_\_\_\_\_\_\_\_\_\_

a) Bug

b) Threat

**c) Vulnerability**

d) Attack

2) Possible threat to any information cannot be \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

a) reduced

b) transferred

c) protected

**d) ignored**

3) A population is a set of \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_of a relation (table) that satisfy some selection condition

a) attributes

**b) tuples**

c) values

d) columns

Exercise

Q.1 What are statistical queries.

Q.2 Explain about statistical database security.

Q.3 Give examples for statistical queries.

**Learning from this lecture**: Learners will be able to understand the concept of statistical database security and give examples.

**Lecture 6**

**Introduction to 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: 1. confidential (C) and 2. Non-confidential (N) and allows all flows except those from class C to class N.

**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. Covert channels can be classified into two broad categories:

* **Storage channels** do not require any temporal synchronization, in that information is conveyed by accessing system information or what is otherwise inaccessible to the user.
* **Timing channel** allow the information to be conveyed by the timing of events or processes.

Some security experts believe that one way to avoid covert channels is for programmers to not actually gain access to sensitive data that a program is supposed to process after the program has been put into operation.

*Let’s check the take away from this lecture*

1) The following is a class for flow policy

**a. confidential**  b. sensitivity

c. privacy d. secrecy

2) ----------------- checks that information contained in some objects does not flow explicitly or implicitly into less protected objects.

a. Flow Policy **b. Flow Control**

b. Data Control d. Organizational Policy

3) A ------------- specifies the channels along which information is allowed to move.

a) security policy

b) privacy policy

**c) flow policy**

d) none of the above

Exercise

Q.1 Explain about flow control security mechanism.

Q.2 Define covert channels.

Q.3 List the categories of covert channels and explain them.

**Learning from this lecture**: Learners will be able to understand the concept of flow control database security and give examples.

**Conclusion**

The study of Database Security helps to know the different security issues and the different security mechanisms to handle those issues. Also the students can understand in detail about the concept of security mechanisms with practical and SQL examples.

**Short Answer Questions:**

1. What is database security?

**Ans:**

Database security refers to the range of tools, controls, and measures designed to establish and preserve database confidentiality, integrity, and availability.

2. Define the term ‘threat’.

**Ans:**

Any situation or event, whether intentionally or incidentally, can cause damage, which can reflect an adverse effect on the database structure and, consequently, the organization. A threat may occur by a situation or event involving a person or the action or situations that are probably to bring harm to an organization and its database.

3. State the difference between authorization and authentication.

**Ans:**

Authentication means confirming your own identity, whereas authorization means being allowed access to the system. In even more simpler terms authentication is the process of verifying oneself, while authorization is the process of verifying what you have access to.

4. What is SQL Injection?

**Ans.**

SQL injection, also known as SQLI, is a common attack vector that uses malicious SQL code for backend database manipulation to access information that was not intended to be displayed. This information may include any number of items, including sensitive company data, user lists or private customer details.

5. List the types of security

**Ans.**

Database security is a broad area that addresses many issues, including the following:

* Various **legal and ethical issues** regarding the right to access certain information—for example, some information may be deemed to be **private** and cannot be accessed legally by unauthorized organizations or persons. In the United States, there are numerous laws governing **privacy of information.**
* **Policy issues** at the governmental, institutional, or corporate level as to what kinds of information should not be made publicly available—for example, **credit ratings** and **personal medical records.**
* **System-related issues** such as the system levels at which various security functions should be enforced—for example, whether a security function should be handled at the physical **hardware level,** the **operating system level,** or the **DBMS level.**

6. List the control measures used for providing security.

**Ans.**

* **Access control:** creating user accounts and passwords to control the login process by the DBMS.
* **Inference control:** Cannot deduce or infer certain facts concerning individuals from the DB.
* **Flow control:** Protect information from reaching unauthorized users.
* **Data encryption:** The data is encoded using some coding algorithm.

7. Compare Discretionary Access Control and Mandatory Access Control.

**Ans.**

**Discretionary access control**

1. Subjects have full control of the objects they have. The discretionary part of 'DAC' means that the file owner has the ability to change the permissions on the file.
2. Uses file permissions and ACL's to restrict access based on users’ identity or group membership.
3. It is more labour intensive and flexible as compared to MAC.
4. All windows, linux, unix and mac os uses DAC.
5. Disadvantage is their weakness to malicious attacks like trojan horses which may appear in application program.

**Mandatory Access Control**

1. An additional security policy that classifies the user and data based on security classes is called MAC.
2. Each subject and object is labelled with a security label.
3. It is less labour intensive and flexible as compared to DAC.
4. MAC based commercial systems are trusted solaries and SE linux.
5. Disadvantage is that being too strict In that they need a firm classification of subjects and objects into security levels and hence they are not applicable to various environments.

8. What is Statistical Database Security?

**Ans.**

Statistical database security focuses on the protection of confidential individual values stored in so called statistical databases and used for statistical purposes. The goal of statistical data protection is to maximize the privacy while minimizing the information loss.

9. What are Covert Channels?

**Ans.**

A covert channel is a type of attack that creates a capability to transfer information objects between processes that are not supposed to be allowed to communicate by the computer security policy.

**Long Answer Questions:**

1. Explain Discretionary Access Control.

**Ans:** Refer the contents under lecture 1

2. Explain Mandatory Access Control with example.

**Ans:** Refer the topic ‘Mandatory Access Control’ under lecture 2

3. Explain Role-Based Access Control.

**Ans:** Refer the topic ‘Role-Based Access Control’ under lecture 2

4. Illustrate with an Example for Granting and Revoking of Privileges.

**Ans:** Referthe section ‘An Example to Illustrate Granting and Revoking of Privileges’ under lecture 2

5. Explain vertical and horizontal propagation with examples?

**Ans:** Refer the contents under lecture 2

6. Describe SQL Injection in detail.

**Ans.** Refer the contents under lecture 4

7. What is flow control security mechanism? Explain in detail.

**Ans.** Refer the contents under lecture 6

8. Explain about Statistical Database Security.

**Ans.** Refer the contents under lecture 5

**Set of Questions for FA/IA/ESE**

1. Explain Discretionary Access Control. (10M)

2. State the difference between authorization and authentication (5M)

3. List the types of security and explain briefly. (5M)

4. Compare Discretionary Access Control and Mandatory Access Control. (5M)

5. State the role of DBA. (5M)

6. Write short notes on Role-Based Access Control. (10M)

7. Explain the protection techniques against SQL Injection (10 M)

8. What is flow control security mechanism? Explain in detail. (10 M)

9. Explain about Statistical Database Security. (10 M)

10. Explain vertical and horizontal propagation with examples. (5 M)

11. Illustrate with an Example for Granting and Revoking of Privileges. (10 M)

**References:**

1. “Fundamentals of Database Systems”, Elmasri & Navathe PEARSON Education, Seventh Edition 2016.

**Self-assessment**

Q.1) What is Database Security? What is the need for providing data security mechanisms.

Q.2) Explain Discretionary Access Control.

Q.3) Explain Mandatory and Role-Based Access Control.

Q.4) Explain the need for multi level security.

Q.5) Explain SQL injection and the different protection techniques used to handle SQL injection.

Q.6) Describe about Statistical database security and flow control database security.

**Self-evaluation**

|  |  |  |
| --- | --- | --- |
| Name of Student |  | |
| Class |  | |
| Roll No. |  | |
| Subject |  | |
| Module No. |  | |
| S.No |  | Tick  Your choice |
|  | Do you understand the need for database security? | * Yes * No |
|  | Do you understand the concept of different access control measures? | * Yes * No |
|  | Do you understand the impact of SQL injection and the protection techniques used to handle it? | * Yes * No |
|  | Do you understand the concept of Statistical database security? | * Yes * No |
|  | Do you understand the concept of flow control database security? | * Yes * No |
|  | Do you understand module 1 ? | * Yes, Completely. * Partialy. * No, Not at all. |