Read Chapter 30

Database Security

Outlines

1. Introduction to Database Security Issues
2. Discretionary Access Control Based on Granting and Revoking Privileges
3. Mandatory Access Control and Role-Based Access Control for Multilevel Security
4. SQL Injection
5. Introduction to Statistical Database Security
6. Introduction to Flow Control
7. Encryption and Public Key Infrastructures
8. Privacy Issues and Preservation
9. Challenges to Maintaining Database Security

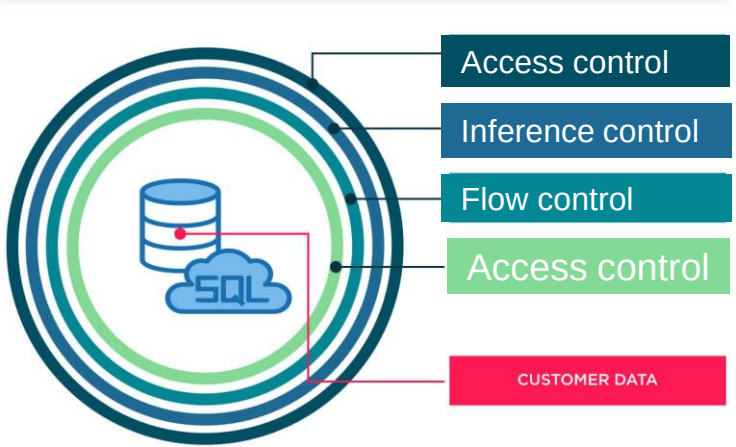
1 Introduction to Database Security Issues

Introduction to Database Security Issues

- Database security a broad area
  - Legal, ethical, policy, and system-related issues
- Threats to databases
  - Loss of integrity
    - Improper modification of information
  - Loss of availability
    - Legitimate user cannot access data objects
  - Loss of confidentiality
    - Unauthorized disclosure of confidential information

Introduction to Database Security Issues (cont'd.)

- Database works as part of a network of services
  - Applications, Web servers, firewalls, SSL terminators, and security monitoring systems
- Types of database control measures
  - Access control
  - Inference control
  - Flow control
  - Encryption



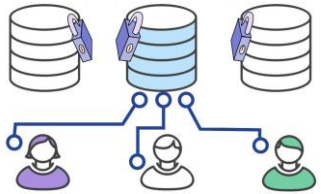
Introduction to Database Security Issues (cont'd.)

- **Discretionary security mechanisms**
  - Used to grant privileges to users
- **Mandatory security mechanisms**
  - Classify data and users into various security classes
  - Implement security policy
- **Role-based security**

Introduction to Database Security Issues (cont'd.)

- Control measures
  - **Access control**

    - Handled by creating user accounts and passwords
  - **Inference control**
    - Must ensure information about individuals cannot be accessed
  - **Flow control**
    - Prevents information from flowing to unauthorized users
  - **Data encryption**
    - Used to protect sensitive transmitted data



Database Security and the DBA

- Database administrator (DBA)
  - Central authority for administering database system
  - **Superuser** or system account
- DBA-privileged commands
  - Account creation
  - Privilege granting
  - Privilege revocation
  - Security level assignment

Access Control, User Accounts, and Database Audits

- User must log in using assigned username and password
- **Login session**
  - Sequence of database operations by a certain user
  - Recorded in system log
- **Database audit**
  - Reviewing log to examine all accesses

  and operations applied during a certain time period

Sensitive Data and Types of Disclosures

- Sensitivity of data: Several factors can cause data to be classified as sensitive:
  - Inherently sensitive (medical record)
  - From a sensitive source
  - Declared sensitive
  - A sensitive attribute or sensitive record
  - Sensitivity in relation to previously disclosed data

Sensitive Data and Types of Disclosures (cont'd.)

- Factors in deciding whether it is safe to reveal the data
  - **Data availability**
    - Not available when being updated (temporary blocking)
  - **Access acceptability**
    - Authorized users
  - **Authenticity assurance**
    - External characteristics of the user
    - Example: access only allowed during working hours

Sensitive Data and Types of Disclosures (cont'd.)

- Typically a tradeoff between precision and security
- **Precision**
  - Protect all sensitive data while making available as much nonsensitive data as possible
- **Security**
  - Ensuring data kept safe from **corruption** and access suitably controlled

Relationship Between Information Security and Information Privacy

- Concept **of privacy goes beyond security:** 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.
  - Ability of individuals to control the terms under which their personal information is acquired and used
  - Security a required building block for privacy
- Preventing storage of personal information
- Ensuring appropriate use of personal information
- Trust relates to both security and privacy

2 Discretionary Access Control Based on Granting and Revoking Privileges

Discretionary Access Control Based on Granting and Revoking Privileges

- Two levels for **assigning privileges** to use a database system
  - Account level
    - Example: CREATE SCHEMA or CREATE TABLE privilege
    - Not defined for SQL2
  - Relation (or table) level
    - Defined for SQL2
    - Access matrix model

Discretionary Access Control (cont'd.)

- Relation or table level (cont'd.)
  - Each relation R assigned an **owner account**
  - Owner of a relation given all privileges on that relation
  - Owner can **grant** privileges to other users on any owned relation
    - SELECT (retrieval or read) privilege on R
    - Modification privilege on R
    - References privilege on R

Specifying Privileges Through the Use of Views

- Consider owner A of relation R and other party B
  - A can create **view V** of R that includes only attributes A wants B to access
    - **Grant SELECT on V to B**
- Can define the view with a query that selects only those tuples from R that A wants B to access

Revocation and Propagation of Privileges

- **Revoking of Privileges**
  - Useful for granting a privilege temporarily
  - **REVOKE** command used to cancel a privilege
- **Propagation of privileges** using the **GRANT OPTION**
  - If GRANT OPTION is given, B can grant privilege to other accounts
  - DBMS must keep track of how privileges were granted if DBMS allows propagation

Example

- 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;
- 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.
- Suppose that A1 creates the two base relations EMPLOYEE and DEPARTMENT, A1 is then the **owner** of these two relations and hence has *all the relation privileges* on each of them.

EMPLOYEE

Name	Ssn	Bdate	Address	Sex	Salary	Dno

DEPARTMENT

Dnumber	Dname	Mgr_ssn

Example

- 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 command:
  - **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.
- 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.

Example

- 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.

Example

- 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;

Example

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

Revocation and Propagation of Privileges (cont'd.)

- **Horizontal and vertical propagation limits**
  - Limiting **horizontal propagation** to an integer number *i*
    - Account *B* given the GRANT OPTION can grant the privilege to at most *i* other accounts
  - **Vertical propagation** limits the depth of the granting of privileges
  - Not available currently in SQL or other relational systems

Revocation and Propagation of Privileges (example)

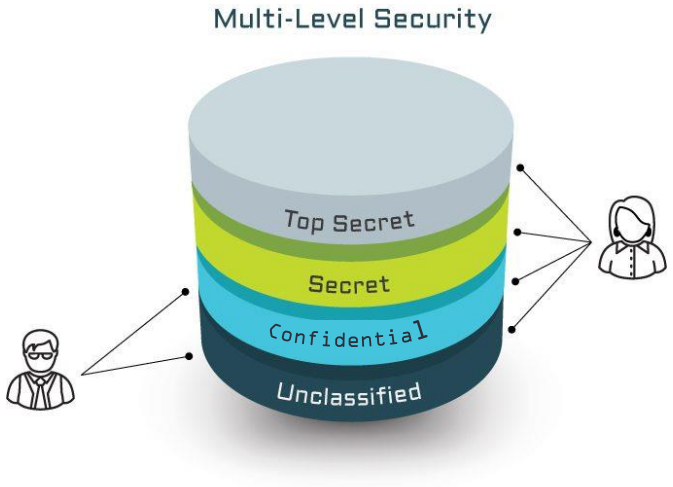
- Suppose that A1 grants SELECT to A2 on the EMPLOYEE relation with **horizontal propagation equal to 1** and **vertical propagation equal to 2**. A2 can then grant SELECT to at most one account because the horizontal propagation limitation is set to 1.
- Additionally, A2 cannot grant the privilege to 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 are designed to limit the depth and breadth of propagation of privileges.

3

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

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

- Mandatory access control
  - Additional security policy that classifies data and users based on security classes
  - Typical security classes
    - Top secret
    - Secret
    - Confidential
    - Unclassified
  - Bell-LaPadula model
    - Subject and object classifications



Mandatory Access Control and Role-Based Access Control for Multilevel Security (cont'd.)

- **Simple security property**
  - Subject S not allowed **read** access to object O unless

  class (S) ≥ class (O)
- **Star property**
  - Subject not allowed to **write** an object unless **class (S) ≤ class (O)**
  - Prevent information from flowing from higher to lower classifications
- Attribute values and tuples considered as data objects

MAC and RBAC for Multilevel Security (cont'd.)

- In a multilevel model, attribute values and tuples considered as data objects
  - Attribute A is associated with a classification attribute C in the schema
  - Each attribute value in a tuple is associated with a corresponding security classification.
  - In some models, a **tuple classification** attribute TC is added to the relation attributes to provide a classification for each tuple as a whole.
  - 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 C_i represents the classification attribute associated with attribute A_i.
  - The value of TC in each tuple t—which is the highest of all attribute classification values within t—provides a general classification for the tuple itself.

MAC and RBAC for Multilevel Security (cont'd.)

- 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 clearance levels.

Mandatory Access Control and Role-Based Access Control for Multilevel Security (cont'd.)

(a) EMPLOYEE

Name	Salary	JobPerformance	TC
Smith U	40000 C	Fair S	S
Brown C	80000 S	Good C	S

(b) EMPLOYEE

Name	Salary	JobPerformance	TC
Smith U	40000 C	NULL C	C
Brown C	NULL C	Good C	C

(c) EMPLOYEE

Name	Salary	JobPerformance	TC
Smith U	NULL U	NULL U	U

(d) EMPLOYEE

Name	Salary	JobPerformance	TC
Smith U	40000 C	Fair S	S
Smith U	40000 C	Excellent C	C
Brown C	80000 S	Good C	S

Figure 30.2 A multilevel relation to illustrate multilevel security (a) The original EMPLOYEE tuples (b) Appearance of EMPLOYEE after filtering for classification C users (c) Appearance of EMPLOYEE after filtering for classification U users (d) Polyinstantiation of the Smith tuple

MAC and RBAC for Multilevel Security (cont'd.)
Example of Figure 30.2

- Assume that the Name attribute is the apparent key, and consider the query SELECT * FROM EMPLOYEE.
- A user with **security clearance S** would see the same relation shown in Figure 30.2(a), since all tuple classifications are less than or equal to S.
- A user with **security clearance C** would not be allowed to see the values for Salary of 'Brown' and Job_performance of 'Smith', since they have higher classification. The tuples would be filtered to appear as shown in Figure 30.2(b), with Salary and Job_performance appearing as null.
- For a user with **security clearance U**, the filtering allows only the Name attribute of 'Smith' to appear, with all the other attributes appearing as null (Figure 30.2(c)). Thus, filtering introduces **null** values for attribute values whose security classification is higher than the user's security clearance.

Figure 30.2 A multilevel relation to illustrate multilevel security (a) The original EMPLOYEE tuples (b) Appearance of EMPLOYEE after filtering for classification C users (c) Appearance of EMPLOYEE after filtering for classification U users (d) Polyinstantiation of the Smith tuple

(a) **EMPLOYEE**

Name	Salary	JobPerformance	TC
Smith U	40000 C	Fair S	S
Brown C	80000 S	Good C	S

(b) **EMPLOYEE**

Name	Salary	JobPerformance	TC
Smith U	40000 C	NULL C	C
Brown C	NULL C	Good C	C

(c) **EMPLOYEE**

Name	Salary	JobPerformance	TC
Smith U	NULL U	NULL U	U

(d) **EMPLOYEE**

Name	Salary	JobPerformance	TC
Smith U	40000 C	Fair S	S
Smith U	40000 C	Excellent C	C
Brown C	80000 S	Good C	S

MAC and RBAC for Multilevel Security (cont'd.)
Example of Figure 30.2

- 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.
- Additionally, 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.
- Other integrity rules, called null integrity and interinstance 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.

Figure 30.2 A multilevel relation to illustrate multilevel security (a) The original EMPLOYEE tuples (b) Appearance of EMPLOYEE after filtering for classification C users (c) Appearance of EMPLOYEE after filtering for classification U users (d) Polyinstantiation of the Smith tuple

(a) EMPLOYEE

Name	Salary	JobPerformance	TC
Smith U	40000 C	Fair S	S
Brown C	80000 S	Good C	S

(b) EMPLOYEE

Name	Salary	JobPerformance	TC
Smith U	40000 C	NULL C	C
Brown C	NULL C	Good C	C

(c) EMPLOYEE

Name	Salary	JobPerformance	TC
Smith U	NULL U	NULL U	U

(d) EMPLOYEE

Name	Salary	JobPerformance	TC
Smith U	40000 C	Fair S	S
Smith U	40000 C	Excellent C	C
Brown C	80000 S	Good C	S

MAC and RBAC for Multilevel Security (cont'd.): Example of Figure 30.2

- To illustrate polyinstantiation, suppose that a user with **security clearance C** tries to update the value of Job_performance of 'Smith' to 'Excellent';
- Since the view provided to users with security clearance C (see Figure 30.2(b)) permits such an update, the system **should not reject it;** otherwise, the user could infer that some **nonnull** value exists for the Job_performance attribute of 'Smith' rather than the null value that appears. This is an example of inferring information through what is known as a covert channel, which should not be permitted in highly secure systems. However, the user should not be allowed to overwrite the existing value of Job_performance at the higher classification level.
- The solution is to create a polyinstantiation for the 'Smith' tuple at the lower classification level C, as shown in Figure 30.2(d). This is necessary since the new tuple cannot be filtered from the existing tuple at classification S.
- The basic update operations of the relational model (INSERT, DELETE, UPDATE) must be modified to handle this and similar situations, but this aspect of the problem is outside the scope of this course.

Figure 30.2 A multilevel relation to illustrate multilevel security (a) The original EMPLOYEE tuples (b) Appearance of EMPLOYEE after filtering for classification C users (c) Appearance of EMPLOYEE after filtering for classification U users (d) Polyinstantiation of the Smith tuple

(a) **EMPLOYEE**

Name	Salary	JobPerformance	TC
Smith U	40000 C	Fair S	S
Brown C	80000 S	Good C	S

(b) **EMPLOYEE**

Name	Salary	JobPerformance	TC
Smith U	40000 C	NULL C	C
Brown C	NULL C	Good C	C

(c) **EMPLOYEE**

Name	Salary	JobPerformance	TC
Smith U	NULL U	NULL U	U

(d) **EMPLOYEE**

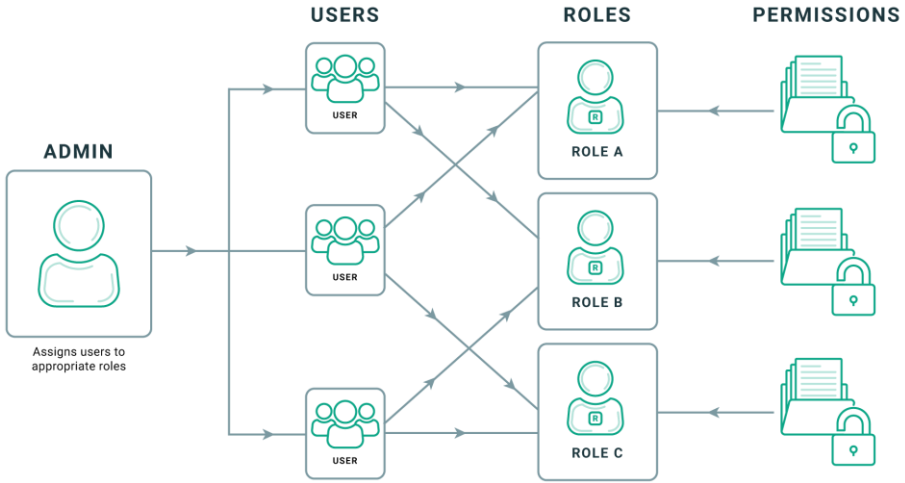
Name	Salary	JobPerformance	TC
Smith U	40000 C	Fair S	S
Smith U	40000 C	Excellent C	C
Brown C	80000 S	Good C	S

Comparing Discretionary Access Control and Mandatory Access Control

- DAC policies have a high degree of flexibility
  - Do not impose control on how information is propagated
- Mandatory policies ensure high degree of protection
  - Rigid
  - Prevent illegal information flow

Role-Based Access Control

- Permissions associated with organizational roles
  - Users are assigned to appropriate roles
- Can be used with traditional **discretionary and mandatory access control**
- Mutual exclusion of roles
  - Authorization time exclusion
  - Runtime exclusion
- Identity management



Label-Based Security and Row-Level Access Control

- Sophisticated access control rules implemented by considering the data **row by row**
- Each row given a label
  - Used to prevent unauthorized users from viewing or altering certain data
- Provides finer granularity of data security
- Label security policy
  - Defined by an administrator

XML Access Control

- Digital signatures for XML
  - XML Signature Syntax and Processing specification
    - Defines mechanisms for countersigning and transformations
- XML encryption
  - XML Encryption Syntax and Processing specification
    - Defines XML vocabulary and processing rules

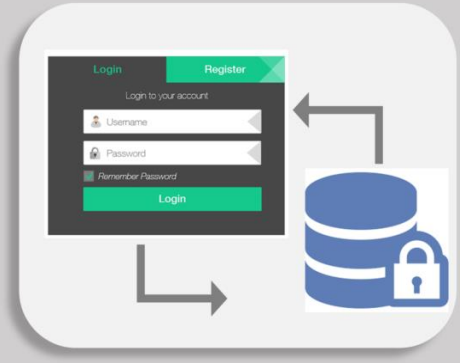
Access Control Policies for the Web and Mobile Applications

- E-commerce environments require elaborate access control policies
  - Go beyond traditional DBMSs
- Legal and financial consequences for unauthorized data breach
- Content-based access control
  - Takes protection object content into account
- Credentials

4

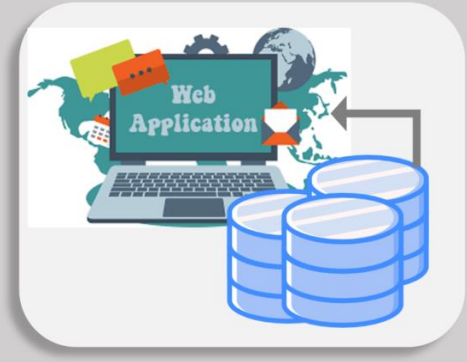
SQL Injection

SQL injection



An SQL query, is more of an application interacting with database

An SQL Injection occurs when an application fails to sanitize the user input data



An attacker can use specially crafted SQL commands to control web application's database server

SQL Injection

- SQL injection
  - Most common threat to database system
- Other common threats
  - Unauthorized privilege escalation
  - Privilege abuse
  - Denial of service
  - Weak authentication

SQL Injection Methods

- Attacker injects a string input through the application
  - Changes or manipulates SQL statement to attacker's advantage
- Unauthorized data manipulation or execution of system-level commands
- SQL manipulation
  - Changes an SQL command in the application
  - Example: adding conditions to the WHERE clause

SQL Injection Methods (cont'd.)

- SQL manipulation (cont'd.)
  - Typical manipulation attack occurs during database login

SELECT * FROM users WHERE username = 'jake' and PASSWORD = 'jakespasswd' ;

    - The attacker can try to manipulate the SQL statement by changing it as follows:

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

    - 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.
- Code injection
  - Add additional SQL statements or commands that are then processed
- Function call injection
  - Database or operating system function call inserted into vulnerable SQL statement to manipulate data or make a privileged system call

Risks Associated with SQL Injection

- 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
  - the attacker can gain access to the database as an authorized user and perform all the desired tasks.
- Identifying injectable parameters
  - the attacker gathers important information about the type and structure of the back-end database of a Web application (the default error page returned by application servers is often overly descriptive).
- Executing remote commands
  - For example, a remote user can execute stored database procedures and functions from a remote SQL interactive interface.
- Performing privilege escalation
  - The attack upgrades the access level.

Protection Techniques

- Blind variables (using parameterized statements)
  - Protects against injection attacks
  - 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);
```

- Filtering input (input validation)
  - Remove escape characters from input strings
  - Escape characters can be used to inject manipulation attacks
- Function security
  - Standard and custom functions should be restricted

5

Introduction to Statistical Database Security

Introduction to Statistical Database Security

- Statistical databases used to provide statistics about various populations
  - Users permitted to retrieve statistical information
  - Must prohibit retrieval of individual data
- Population: set of tuples of a relation (table) that satisfy some selection condition

PERSON

Name	Ssn	Income	Address	City	State	Zip	Sex	Last_degree

Figure 30.3 The PERSON relation schema for illustrating statistical database security

Introduction to Statistical Database Security (cont'd.)

```
Q1: SELECT COUNT (*)FROM PERSON
    WHERE <condition>;
Q2: SELECT AVG (Income) FROM PERSON
    WHERE <condition>;
```

- Only statistical queries are allowed
- Preventing the inference of individual information
  - Provide minimum threshold on number of tuples
  - Prohibit sequences of queries that refer to the same population of tuples
  - Introduce slight noise or inaccuracy
  - Partition the database
    - Store records in groups of minimum size

6

Introduction to Flow Control

Introduction to Flow Control

- **Flow control**
  - Regulates the distribution or flow of information among accessible objects
  - Verifies information contained in some objects does not flow explicitly or implicitly into less protected objects
- **Flow policy**
  - Specifies channels along which information is allowed to move
    - Simple form: confidential (C) and non-confidential (N)
    - Allows all flows except those from class *C* to class *N*.

Introduction to Flow Control (cont'd.)

- **Covert channels**
  - Allows information to pass from a higher classification level to a lower classification level through improper means
  - classified into two broad categories:
    - Timing channel requires temporal synchronization
    - Storage channel does not require temporal synchronization
- A simple example of a covert channel, consider a distributed database system in which two nodes have user security levels of secret (S) and unclassified (U).
  - In order for a transaction to commit, **both nodes must agree** to commit.

7

Encryption and Public Key Infrastructures

Encryption and Public Key Infrastructures

- **Encryption converts data into cyphertext**
  - Performed by applying an encryption algorithm to data using a prespecified encryption key
  - Resulting data must be decrypted using a decryption key to recover original data
- **Data Encryption Standard (DES)**
  - Developed by the U.S. Government for use by the general public
- **Advanced Encryption Standard (AES)**
  - More difficult to crack

Encryption and Public Key Infrastructures (cont'd.)

- **Symmetric key algorithms**
  - Also called secret key algorithms
  - Need for sharing the secret key
    - Can apply some function to a user-supplied password string at both sender and receiver
- **Public (asymmetric) key encryption**
  - Involves public key and private key
  - Private key is not transmitted
  - Two keys related mathematically
    - Very difficult to derive private key from public key

Encryption and Public Key Infrastructures (cont'd.)

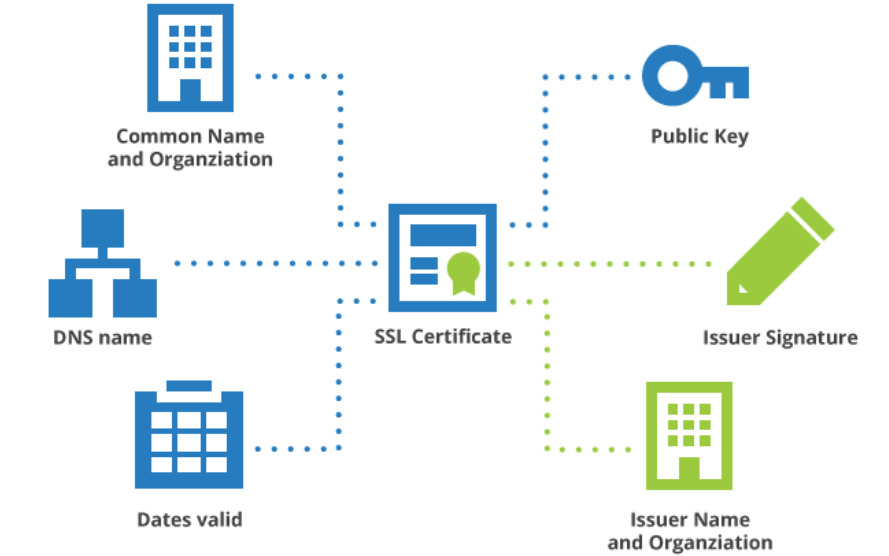
- **Public (asymmetric) key encryption steps**
  - Each user generates a pair of keys to be used for encryption and decryption of messages
  - Each user places public key **in a public register or other accessible file**
    - Keeps companion key private
  - Sender encrypts message using receiver's public key
  - Receiver decrypts message using receiver's private key
- RSA public key encryption algorithm

Digital Signatures

- A digital signature **is an example of using encryption** techniques to provide authentication services in electronic commerce applications.
- Consist of string of symbols
- Each is unique and signatures must be different for **each use**
  - Function of the message it is signing, along with a timestamp
  - Depends on secret number unique to the signer
- **Public key** techniques used to create digital signatures

Digital Certificates

- Combines value of a public key with the identity of the person or service that holds the corresponding private key into a digitally signed statement
- Information included in the certificate
  - Owner information
  - Public key of the owner
  - Date of certificate issue and validity period
  - Issuer identification
  - Digital signature



8 Privacy Issues and Preservation

Privacy Issues and Preservation

Preserving data privacy is a growing challenge for database security

- Limit performing large-scale mining and analysis
  - **Avoid building big central warehouses** for vital information
    - Violating security could expose all data
- Distributed data mining algorithms
  - Remove identity information in released data
  - Inject noise into the data
    - Must be able to estimate errors introduced
- Mobile device privacy

9

Challenges to Maintaining Database Security

Challenges to Maintaining Database Security

- Data quality
  - Quality stamps are posted on Web sites
  - Application-level **recovery techniques** to automatically repair incorrect data
- Intellectual property rights
  - Digital watermarking techniques : **protect** content from unauthorized duplication and distribution by enabling **provable ownership** of the content

Challenges to Maintaining Database Security (cont'd.)

- Database survivability
  - **Confinement :** Take immediate action to eliminate the attacker's access to the system and to isolate or contain the problem to prevent further spread.
  - **Damage assessment:** Determine the extent of the problem, including failed functions and corrupted data.
  - **Reconfiguration:** Reconfigure to allow operation to continue in a **degraded mode** while recovery proceeds.
  - **Repair :** Recover corrupted or lost data and repair or reinstall failed system functions to reestablish a normal level of operation.
  - **Fault treatment :** To the extent possible, identify **the weaknesses** exploited in the attack and take steps to prevent a recurrence.

Oracle Label-Based Security

- Oracle label security
  - Enables row-level access control
  - Every table or view has an associated security policy
- **Virtual private database (VPD) technology**
  - Feature that adds predicates to user statements to limit their access in a transparent manner to the user and the application
  - The VPD concept allows server-enforced, fine-grained access control for a secure application.
  - Based on policies
- Oracle label security is a technique of **enforcing row-level security** in the form of a security policy.

Label Security Architecture

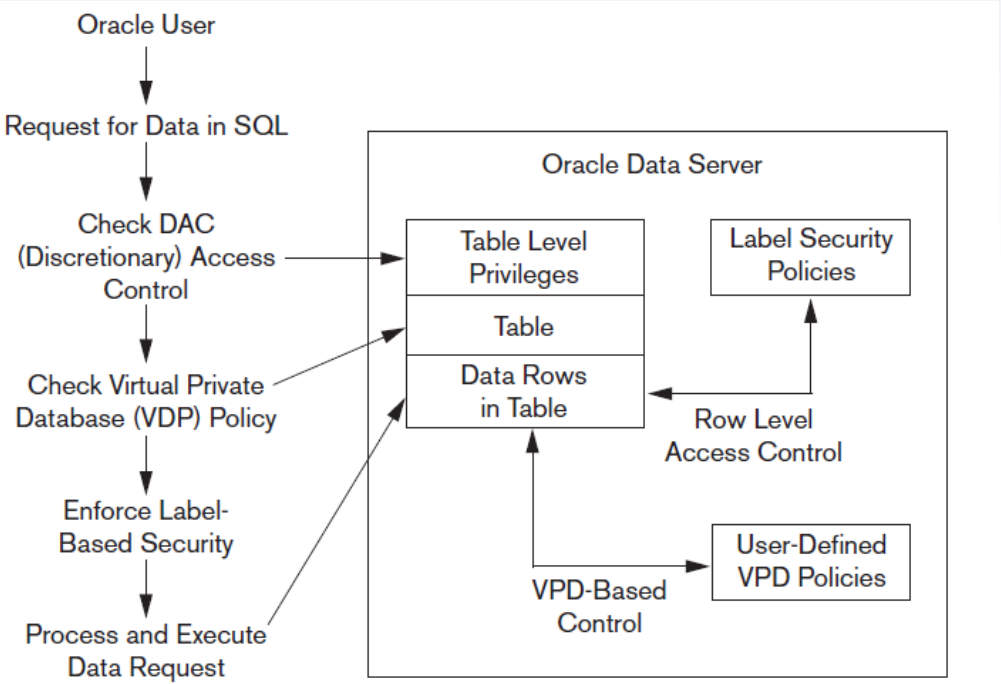


Figure 30.4 Oracle label security architecture. Data from: Oracle (2007)

How Data Labels and User Labels Work Together

- A user's label indicates the information the user is permitted to access. It also determines the type of access **(read or write)** that the user has on that information.
- A row's label shows the sensitivity of the information that the row contains as well as the ownership of the information.
- Compartments allow a finer classification of sensitivity of the labeled data. All data related to the same project can be labeled with the same compartment. Compartments are optional; a label can contain zero or more compartments.
- Groups are used to identify organizations as owners of the data with corresponding group labels. Groups are hierarchical; for example, a group can be associated with a **parent group**.

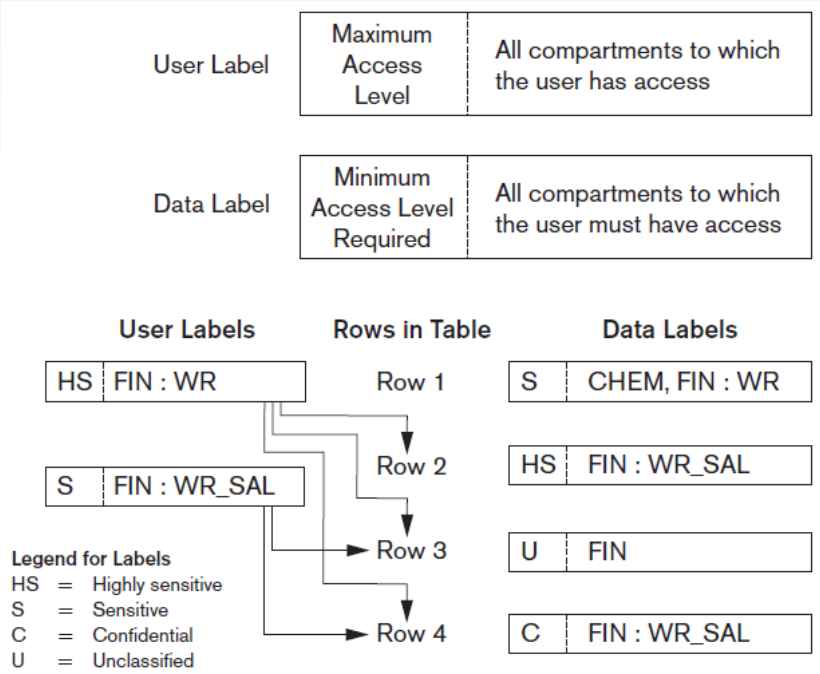


Figure 30.5 Data labels and user labels in Oracle. Data from: Oracle (2007), FIN: Finance; WR:Western Region; WR_SAL:WR Sales

Summary

- Threats to databases
- Types of control measures
  - Access control
  - Inference control
  - Flow control
  - Encryption
- Mandatory access control
- SQL injection
- Key-based infrastructures