

## 04. Database Security

# Database Security

- Relates to DBMS features and measures that protect the DBMS against service degradation and protect the database against loss, corruption or mishandling
- The DBMS should be secured from the point of installation through operation and maintenance

**The database security strategy must lead to a situation where data is protected, reconstructable, auditable, and tamperproof, and users are identifiable, authorized, and monitored**

**James Martin**

# Intro to Database Security

- Database Security a broad area:
  - Legal and ethical issues:
    - Related to the right to access information – some info considered private, other protected even by the law
  - Policy issues:
    - Governmental, Organizational level etc. – what information should be made publicly available?
  - System-related issues
    - What level will security be enforced: hardware level, operating system level, or DBMS level?
  - The need to identify multiple security levels
    - To cater for different users e.g. a classification of info into Top Secret, Secret, Confidential and Unclassified

# Threats to Database Security

- **Loss of integrity**
  - Requirement that data be protected from improper modification – intentional/accidental update, addition, deletion etc.
- **Loss of availability:**
  - Available to human users and programs that have a legitimate right to it
- **Loss of confidentiality:**
  - Protection from unauthorized disclosure – usually results in loss of public confidence, embarrassment or legal action against organization

## Countermeasures:

- **Access Control:**
  - Discretionary Access Control
  - Mandatory Access Control
  - Role-Based Access Control
- **Inference Control**
- **Flow Control**
- **Encryption**

# Introduction: Desirable Features of DBMS

- Techniques to enable individual and groups of users to access selected portions of a database system ... especially important in large databases with many different users in an organization
- 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
  - Mandatory security mechanisms

# Database Security

- The task of setting up, and maintaining database security falls on the DBA working in conjunction with a database security officer who establish a comprehensive data security strategy beginning with a security policy
- Security Policy = a collection of standards, policies and procedures that guarantee the security of a system
- Of great relevance at this point is identification of security vulnerabilities and measures to protect the system against the vulnerabilities

# Database Security and the DBA

- The DBA is the central authority for managing a database system.
- Security-Related Responsibilities:
  - granting privileges to users
  - classifying users and data in accordance with the policy of the organization.
- Controls the **DBA account** in the DBMS, sometimes called a **system** or **superuser account**, which provides powerful capabilities:
  - Account Creation
  - Privilege Granting
  - Privilege Revocation
  - Security Level Assignment

SAMPLE SECURITY VULNERABILITIES AND RELATED PROTECTIVE MEASURES		
SYSTEM COMPONENT	SECURITY VULNERABILITY	SECURITY MEASURES
People	<ul style="list-style-type: none"> <li>The user sets a blank password.</li> <li>The password is short or includes a birth date.</li> <li>The user leaves the office door open all the time.</li> <li>The user leaves payroll information on the screen for long periods of time.</li> </ul>	<ul style="list-style-type: none"> <li>Enforce complex password policies.</li> <li>Use multilevel authentication.</li> <li>Use security screens and screen savers.</li> <li>Educate users about sensitive data.</li> <li>Install security cameras.</li> <li>Use automatic door locks.</li> </ul>
Workstation and servers	<ul style="list-style-type: none"> <li>The user copies data to a flash drive.</li> <li>The workstation is used by multiple users.</li> <li>A power failure crashes the computer.</li> <li>Unauthorized personnel can use the computer.</li> <li>Sensitive data is stored on a laptop computer.</li> <li>Data is lost due to a stolen hard disk or laptop.</li> <li>A natural disaster occurs.</li> </ul>	<ul style="list-style-type: none"> <li>Use group policies to restrict the use of flash drives.</li> <li>Assign user access rights to workstations.</li> <li>Install uninterrupted power supplies (UPSs).</li> <li>Add security locks to computers.</li> <li>Implement a kill switch for stolen laptops.</li> <li>Create and test data backup and recovery plans.</li> <li>Protect the system against natural disasters—use co-location strategies.</li> </ul>



## SAMPLE SECURITY VULNERABILITIES AND RELATED PROTECTIVE MEASURES

SYSTEM COMPONENT	SECURITY VULNERABILITY	SECURITY MEASURES
Operating system	<ul style="list-style-type: none"><li>• Buffer overflow attacks</li><li>• Virus attacks</li><li>• Root kits and worm attacks</li><li>• Denial-of-service attacks</li><li>• Trojan horses</li><li>• Spyware applications</li><li>• Password crackers</li></ul>	<ul style="list-style-type: none"><li>• Apply OS security patches and updates.</li><li>• Apply application server patches.</li><li>• Install antivirus and antispyware software.</li><li>• Enforce audit trails on the computers.</li><li>• Perform periodic system backups.</li><li>• Install only authorized applications.</li><li>• Use group policies to prevent unauthorized installations.</li></ul>
Applications	<ul style="list-style-type: none"><li>• Application bugs—buffer overflow</li><li>• SQL injection, session hijacking, etc.</li><li>• Application vulnerabilities—cross-site scripting, nonvalidated inputs</li><li>• Email attacks—spamming, phishing, etc.</li><li>• Social engineering emails</li></ul>	<ul style="list-style-type: none"><li>• Test application programs extensively.</li><li>• Build safeguards into code.</li><li>• Do extensive vulnerability testing in applications.</li><li>• Install spam filters and antivirus software for email systems.</li><li>• Use secure coding techniques (see <a href="http://www.owasp.org">www.owasp.org</a>).</li><li>• Educate users about social engineering attacks.</li></ul>

## SAMPLE SECURITY VULNERABILITIES AND RELATED PROTECTIVE MEASURES

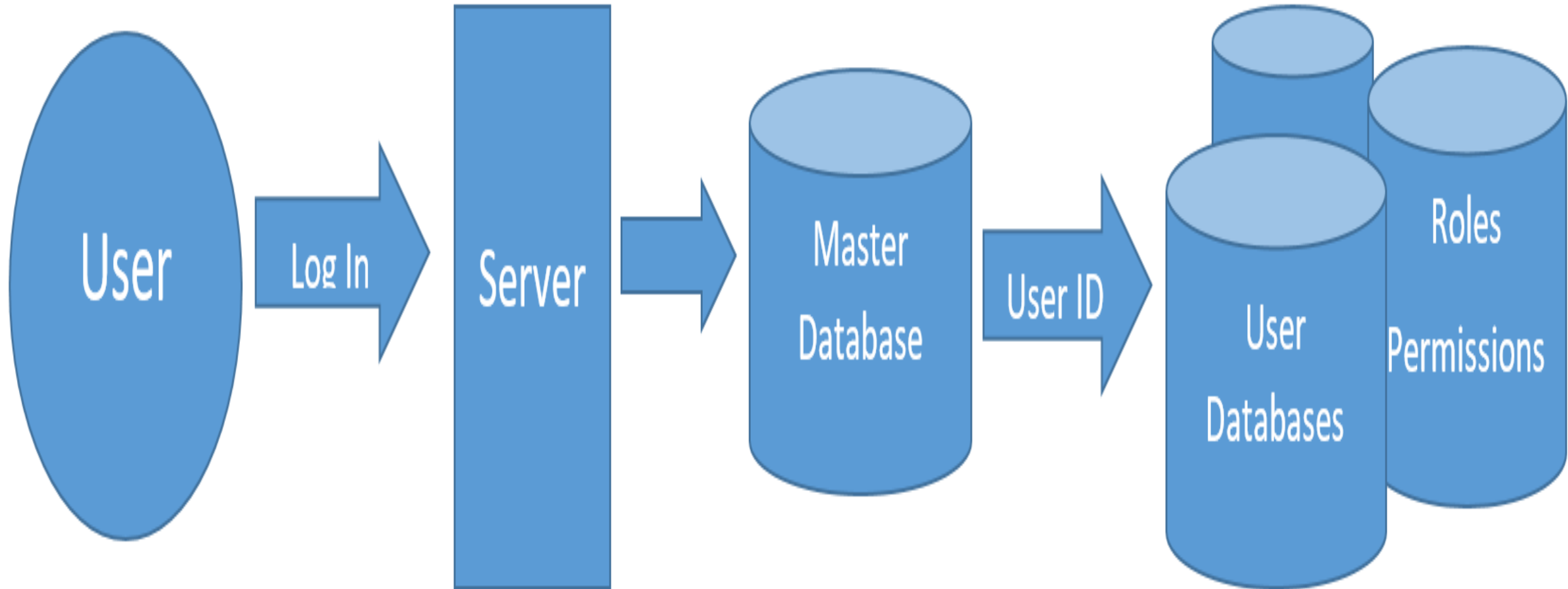
SYSTEM COMPONENT	SECURITY VULNERABILITY	SECURITY MEASURES
Network	<ul style="list-style-type: none"><li>• IP spoofing</li><li>• Packet sniffers</li><li>• Hacker attacks</li><li>• Clear passwords on network</li></ul>	<ul style="list-style-type: none"><li>• Install firewalls.</li><li>• Use virtual private networks (VPNs).</li><li>• Use intrusion detection systems (IDSs).</li><li>• Use network access control (NAC).</li><li>• Use network activity monitoring.</li></ul>
Data	<ul style="list-style-type: none"><li>• Data shares are open to all users.</li><li>• Data can be accessed remotely.</li><li>• Data can be deleted from a shared resource.</li></ul>	<ul style="list-style-type: none"><li>• Implement file system security.</li><li>• Implement share access security.</li><li>• Use access permission.</li><li>• Encrypt data at the file system or database level.</li></ul>



# Other Measures

- Use Access Management
  - User Definition
  - Password Assignment
  - User group Definition
  - Access Privileges Assignment
  - Physical Access Control
- View Definition
- DBMS Access Control
- DBMS Usage Monitoring
- Data Encryption
- Server Separation
- Firewalls
- Scheduled Backups (regular)
- Using up-to-date apps

# Simplified Security Model



# Discretionary Access Control (DAC)

## Types:

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

## DAC - Account Level Privileges:

- |                 |          |
|-----------------|----------|
| • CREATE SCHEMA | • DROP   |
| • CREATE TABLE  | • MODIFY |
| • CREATE VIEW   | • SELECT |
| • ALTER         |          |

# DAC: Relation Level

- Applies to base relations (tables) and virtual relations (views / stored queries)
- Granting and revoking of privileges follow the access matrix model, where rows of matrix represent subjects (users, accounts, programs) and 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  $j$ .

Subjects	Objects					
		Relations	Records	Columns	Views	Operations
	Users					
	Accounts					
	Programs					

# Discretionary Privileges

- Each relation in database is owned by an **owner account**, – typically the account that was used when the relation was created in the first place.
- The owner account holder can pass privileges on any of the owned relation to other users by **granting** privileges to their accounts.

# Specifying Privileges Using Views

- The mechanism of views is an important discretionary authorization mechanism in its own right.
- Examples:
  - Owner A of a relation R wants another account B to be able to retrieve only some fields of R. He creates a view V of R that includes only those attributes and then grant SELECT on V to B.
  - Owner A wishes to limit B to retrieving only certain tuples of R; He creates a view V' by means of a query that selects only those tuples from R that A wants to allow B to access.



# Multilevel Security: Mandatory Access Control

- The discretionary access control techniques of granting and revoking privileges on relations has 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, and *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.

# Multilevel Security: Mandatory Access Control

- Based on classification of information into **security classes**.
- Typical arrangement:
  - Top secret (TS),
  - Secret (S),
  - Confidential (C),
  - Unclassified (U),

where TS is the highest level and U the lowest:  $TS \geq S \geq C \geq U$

- Each **subject** (user, account, program) and **object** (relation, tuple, column, view, operation) is classified into one of the security classifications: T, S, C, or U:

# Multilevel Security: Mandatory Access Control

- Two restrictions are enforced on data access based on the subject/object classifications:
  1. A subject  $S$  is not allowed read access to an object  $O$  unless  $\text{class}(S) \geq \text{class}(O)$ . This is known as the **simple security property**.
    - No subject can read an object whose security classification is higher than that of the subject's security clearance
  2. A subject  $S$  is not allowed to write an object  $O$  unless  $\text{class}(S) \leq \text{class}(O)$ . This known as the **star property** (or  $*$  property).
    - Prevents subject writing to an object of lower security classification. Violation of this rule would allow information flow from higher to lower classification ... violating a basic tenet of multi-level security

# DAC vs. MAC

- 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.
  - Because DAC doesn't restrict / control how information is propagated and used once accessed by authorized users
- 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.

# Multilevel Security: Role-Based Access Control

- Role-based access control (RBAC) have emerged as a proven technology for managing and enforcing security in large-scale enterprise-wide systems.
- Basic notion: Permissions be associated with roles, users are assigned to appropriate roles.
- Use SQL commands CREATE ROLE and DESTROY ROLE. Also GRANT and REVOKE commands to assign and revoke privileges from roles.

# Multilevel Security: Role-Based Access Control

- 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.
  - Role hierarchy in RBAC seems a natural way of organizing roles to reflect the organization's lines of authority and responsibility.
- Using an RBAC model is highly desirable goal for addressing the key security requirements of Web-based applications.
- RBAC can represent traditional DAC and MAC policies as well as other user-defined organizational specific policies

# 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: COUNT, SUM, MIN, MAX, AVERAGE, STANDARD DEVIATION etc. – **Statistical Queries**

# Statistical Database Security

- Example: To retrieve:
  - the number of individuals in a population
  - the average income in the population.
- Important: Ability to restrict statistical users from retrieving individual data, such as the income of a specific person.
- 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.



# Statistical Database Security

- Example:

```
Select Count(*) From Person Where <Condition>
```

```
Select Average(Income) From Person Where <Condition>
```

- Condition

- The Condition phrase can be 'played around with' to the point where one infers information about an individual

- E.g. if the condition is

```
LastQualification = 'PhD' And Gender= 'F' And Age<40
```

# Statistical Database Security – Solutions

- Refusal of queries that return small numbers of records
- Prohibit sequences of queries that refer repeatedly to the same population
- Deliberate introduction of slight inaccuracies or noise to statistical database queries

# Flow Control

- Flow control aims to regulate 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.

# Encryption and Public Key Infrastructures

- Encryption is a means of maintaining secure data in an insecure environment.
- Encryption consists of applying an **encryption algorithm** to data using some prespecified **encryption key**.
  - the resulting data has to be **decrypted** using a **decryption key** to recover the original data.
- A number of standards have been developed to enhance the technique:
  - The Data Encryption Standard
  - Advanced Encryption Standard
  - Public Key Encryption

# Public Key Encryption

- Public key algorithms are based on mathematical functions rather than operations on bit patterns.
  - They also involve the use of two separate but mathematically related keys: the **public key** and the **private key**.
  - the private key is kept secret
  - Public key is made available in a public register or other accessible file

# Public Key Encryption Ingredients:

- *Plaintext* : This is the data or readable message that is fed into the algorithm as input.
- *Encryption algorithm* : The encryption algorithm performs various transformations on the plaintext.
- *Public and private keys* : If one is used for encryption, the other is used for decryption.
- *Ciphertext* : This is the scrambled message produced as output. It depends on:
  - the plaintext
  - the key.
- *Decryption algorithm* : This algorithm accepts the ciphertext and the matching key and produces the original plaintext.

# Public Key Encryption

- Public key is made for public and private key is known only by owner.
- If a sender wishes to send a private message to a receiver, the sender encrypts the message using the receiver's public key.
- When the receiver receives the message, he decrypts it using his private key. No other recipient can decrypt the message because only the receiver knows his private key.

# Digital Signatures

- Signature = a string of symbols that identify author
- If same signature is used all the time, it may become easy / possible to counterfeit
- Digital signature must be different for each use. This is achieved by use of a function that depends on the message content, time stamp and a secret number only known to the signer
- Can be easily implemented within a public key technique