Chapter 9: Database Security: An Introduction

Nguyen Thi Ai Thao
thaonguyen@cse.hcmut.edu.vn

Spring- 2016
Outline

- Introduction to Database Security Issues
  - Types of Security
  - Threats to databases
  - Database Security and DBA
  - Access Protection, User Accounts, and Database Audits

- Discretionary Access Control
  - Types of Discretionary Privileges
  - Specifying Privileges Using Views
  - Revoking Privileges
  - Propagation of Privileges Using the GRANT OPTION
  - An example
  - Weakness

- Mandatory Access Control
  - Bell-LaPudula Model
  - Comparing DAC and MAC

- RBAC (Role-Based Access Control)

- Encryption & PKI (Public Key Infrastructure)
Introduction to Database Security Issues

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

- **Authentication**: a mechanism that determines whether a user is who he or she claims to be.
- **Authorization**: the granting of a right or privilege, which enables a subject to legitimately have access to a system or a system’s objects.
- **Access Control**: a security mechanism (of a DBMS) for restricting access to a system’s objects (the database) as a whole.
Introduction to Database Security Issue(2)

Threats

- Any situation or event, whether intentional or unintentional, that will adversely affect a system and consequently an organization

- Threats to:
  - Computer systems
  - Databases
Threats to Computer Systems

**Hardware**
- Fire/flood/bombs
- Data corruption due to power loss or surge
- Failure of security mechanisms giving greater access
- Theft of equipment
- Physical damage to equipment
- Electronic interference and radiation

**DBMS and Application Software**
- Failure of security mechanism giving greater access
- Program alteration
- Theft of programs

**Communication networks**
- Wire tapping
- Breaking or disconnection of cables
- Electronic interference and radiation

**Database**
- Unauthorized amendment or copying of data
- Theft of data
- Data corruption due to power loss or surge

**Users**
- Using another person’s means of access
- Viewing and disclosing unauthorized data
- Inadequate Staff training
- Illegal entry by hacker
- Blackmail
- Introduction of viruses

**Programers/Operators**
- Creating trapdoors
- Program alteration (such as creating software that is insecure)
- Inadequate staff training
- Inadequate security policies and procedures
- Staff shortages or strikes

**Data/Database Administrator**
- Inadequate security policies and procedures
Scope of Data Security Needs

• Must protect databases & the servers on which they reside
• Must administer & protect the rights of internal database users
  • Must guarantee the confidentiality of ecommerce customers as they access the database
• With the Internet continually growing, the threat to data traveling over the network increases exponentially
Introduction to Database Security Issues (3)

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

- To protect databases against these types of threats, four kinds of countermeasures can be implemented:
  - Access control
  - Inference control
  - Flow control
  - Encryption
Introduction to Database Security Issues (4)

- A DBMS typically includes a database security and authorization subsystem that is responsible for ensuring the security portions of a database against unauthorized access.

- Two types of database security mechanisms:
  - Discretionary security mechanisms
  - Mandatory security mechanisms
Introduction to Database Security Issues 5)

The security mechanism of a DBMS must include provisions for restricting access to the database as a whole

- This function is called **access control** and is handled by creating user accounts and passwords to control login process by the DBMS.
Introduction to Database Security Issues (6)

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

  - The countermeasures to statistical database security problem is called inference control measures.
Another security is that of flow control, which prevents information from flowing in such a way that it reaches unauthorized users.

Channels that are pathways for information to flow implicitly in ways that violate the security policy of an organization are called covert channels.
A final security issue is data encryption, which is used to protect sensitive data (such as credit card numbers) that is being transmitted via some type communication network.

The data is encoded using some encoding algorithm.

- An unauthorized user who access encoded data will have difficulty deciphering it, but authorized users are given decoding or decrypting algorithms (or keys) to decipher data.
Database Security and the DBA

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

- The DBA is responsible for the overall security of the database system.
The DBA has a DBA account in the DBMS
- Sometimes these are called a system or superuser account
- These accounts provide powerful capabilities such as:
  1. Account creation
  2. Privilege granting
  3. Privilege revocation
  4. Security level assignment
- Action 1 is access control, whereas 2 and 3 are discretionary and 4 is used to control mandatory authorization
Access Protection, User Accounts, and Database Audits

- Whenever a person or group of persons need to access a database system, the individual or group must first apply for a user account.
  - The DBA will then create a new account id and password for the user if he/she deems there is a legitimate need to access the database.
- The user must log in to the DBMS by entering account id and password whenever database access is needed.
The database system must also keep track of all operations on the database that are applied by a certain user throughout each login session.

- To keep a record of all updates applied to the database and of the particular user who applied each update, we can modify system log, which includes an entry for each operation applied to the database that may be required for recovery from a transaction failure or system crash.
Access Protection, User Accounts, and Database Audits(3)

- If any tampering with the database is suspected, a **database audit** is performed
  - A database audit consists of reviewing the log to examine all accesses and operations applied to the database during a certain time period.

- A database log that is used mainly for security purposes is sometimes called an **audit trail**.
Outline

Introduction to Database Security Issues
- Types of Security
- Threats to databases
- Database Security and DBA
- Access Protection, User Accounts, and Database Audits

Discretionary Access Control
- Types of Discretionary Privileges
- Specifying Privileges Using Views
- Revoking Privileges
- Propagation of Privileges Using the GRANT OPTION
- Examples
- Weakness

Mandatory Access Control
- Bell-LaPudula Model
- Comparing DAC and MAC

RBAC (Role-Based Access Control)

Encryption & PKI (Public Key Infrastructure)
Discretionary Access Control

- User can protect what they own.
- Owner may grant access to other.
- Owner can define the type of access (read/write/execute/…) given to others.

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

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

- **The relation level (or table level):**
  - At this level, the DBA can control the privilege to access each individual relation or view in the database.
The privileges at the account level apply to the capabilities provided to the account itself and can include:

- the CREATE SCHEMA or CREATE TABLE privilege, to create a schema or base relation;
- the CREATE VIEW privilege;
- the ALTER privilege, to apply schema changes such adding or removing attributes from relations;
- the DROP privilege, to delete relations or views;
- the MODIFY privilege, to insert, delete, or update tuples;
- and the SELECT privilege, to retrieve information from the database by using a SELECT query.
The second level of privileges applies to the relation level

- This includes base relations and virtual (view) relations.

The granting and revoking of privileges generally follow an authorization model for discretionary privileges known as the access matrix model where

- The rows of a matrix $M$ represents subjects (users, accounts, programs).
- The columns represent objects (relations, records, columns, views, operations).
- Each position $M(i,j)$ in the matrix represents the types of privileges (read, write, update) that subject $i$ holds on object $j$. 
Types of Discretionary Privileges(4)

To control the granting and revoking of relation privileges, for each relation R in a database:

- The owner of a relation is given **all** privileges on that relation.
- The owner account holder can **pass privileges** on any of the owned relation to other users by **granting** privileges to their accounts.
- The owner account holder can also **take back** the **privileges** by **revoking** privileges from their accounts.
In SQL the following types of privileges can be granted on each individual relation R:

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

- **REFERENCES** privilege on $R$:
  - This gives the account the capability to reference relation $R$ when specifying integrity constraints.
  - The privilege can also be **restricted** to specific attributes of $R$.

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

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

In some cases it is desirable to grant a privilege to a user temporarily. For example,

- The owner of a relation may want to grant the `SELECT` privilege to a user for a specific task and then revoke that privilege once the task is completed.

- Hence, a mechanism for revoking privileges is needed. In SQL, a `REVOKE` command is included for the purpose of canceling privileges.
Propagating Privileges using the GRANT OPTION

- Whenever the owner A of a relation R grants a privilege on R to another account B, privilege can be given to B with or without the GRANT OPTION.

- If the GRANT OPTION is given, this means that B can also grant that privilege on R to other accounts.
  - Suppose that B is given the GRANT OPTION by A and that B then grants the privilege on R to a third account C, also with GRANT OPTION. In this way, 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.
Limiting the horizontal propagation

Diagram:

- A1
- Grant with horizontal propagation: $i = 2$
- A2
- Grant:
  - A3
  - A4
- $i = 2$
Limiting the vertical propagation
An Example

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

  ```sql
  GRANT CREATETAB TO A1;
  ```

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

  ```sql
  CREATE SCHEMA EXAMPLE AUTHORIZATION A1;
  ```
An Example(2)

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

  GRANT INSERT, DELETE ON
  EMPLOYEE, DEPARTMENT TO A2;
### An Example (3)

#### EMPLOYEE

<table>
<thead>
<tr>
<th>Name</th>
<th>Ssn</th>
<th>Bdate</th>
<th>Address</th>
<th>Sex</th>
<th>Salary</th>
<th>Dno</th>
</tr>
</thead>
</table>

#### DEPARTMENT

<table>
<thead>
<tr>
<th>Dnumber</th>
<th>Dname</th>
<th>Mgr_ssn</th>
</tr>
</thead>
</table>

**Figure 23.1**

Schemas for the two relations EMPLOYEE and DEPARTMENT.
Suppose that A1 wants to allow A3 to retrieve information from either of the two tables and also to be able to propagate the SELECT privilege to other accounts.

A1 can issue the command:

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

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

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

- Notice that A4 can’t propagate the SELECT privilege because GRANT OPTION was not given to A4.
An Example(5)

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

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

The DBMS must now automatically revoke the SELECT privilege on EMPLOYEE from A4, too, because A3 granted that privilege to A4 and A3 does not have the privilege any more.
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 create the view:
```sql
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:
```sql
GRANT SELECT ON A3EMPLOYEE TO A3
WITH GRANT OPTION;
```
An Example(7)

• Finally, suppose that A1 wants to allow A4 to update only the SALARY attribute of EMPLOYEE;

• A1 can issue:

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

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

- Unrestricted DAC allows information from an object which can be read by a subject to be written to any other object
  - Bob is denied access to file Y, so he asks cohort Alice to copy Y to X that he can access

- Suppose our users are trusted not to do this deliberately. It is still possible for Trojan Horses to copy information from one object to another.
Trojan horse Example

User Alice

User Bob

User Bob cannot read the file A!

File A

r: Alice; w:Alice

File B

r: Bob; w:Bob
Trojan horse Example

User Alice

Pgm X

User Bob

TH

r: Alice; w: Alice

File A

r: Bob; w: Bob, Alice

File B
Trojan horse Example

User Alice

User Bob

File A

File B

User Bob can read contents of file A copied to file B

r: Alice; w: Alice

r: Bob; w: Bob, Alice
Outline

- Introduction to Database Security Issues
  - Types of Security
  - Threats to databases
  - Database Security and DBA
  - Access Protection, User Accounts, and Database Audits

- Discretionary Access Control
  - Types of Discretionary Privileges
  - Specifying Privileges Using Views
  - Revoking Privileges
  - Propagation of Privileges Using the GRANT OPTION
  - An example
  - Weakness

- Mandatory Access Control
  - Bell-LaPudula Model
  - Comparing DAC and MAC

- RBAC (Role-Based Access Control)

- Encryption & PKI (Public Key Infrastructure)
Mandatory Access Control

- Granting access to the data on the basis of users’ clearance level and the sensitivity level of the data
- Bell-LaPadula’s two principles: no read-up & no write-down secrecy
Bell-LaPudula Model

- Typical **security classes** are top secret (TS), secret (S), confidential (C), and unclassified (U), where TS is the highest level and U is the lowest one: TS > S > C > U
- Two restrictions are enforced on data access based on the subject/object classifications:
  - A subject S is not allowed read access to an object O unless class(S) ≥ class(O). This is known as the **simple security property**
  - A subject S is not allowed to write an object O unless class(S) ≤ class(O). This known as the **star property** (or * property)
Why star property?

User Alice

User Bob

Pgm X

TH

File A

File B

r: Alice; w: Alice

r: Bob; w: Bob, Alice
Why star property?

User Alice

Pgm X

TH

User Bob

r: Alice; w: Alice

File A

Read

r: Bob; w: Bob, Alice

File B

Write

User Bob can read contents of file A
copied to file B
Why star property?
Multilevel relation

- **Multilevel relation:** MAC + relational database model
- **Data objects:** attributes and tuples
- Each attribute $A$ is associated with a classification attribute $C$
- A **tuple classification** attribute $TC$ is to provide a classification for each tuple as a whole, the highest of all attribute classification values.

\[ R(A_1,C_1,A_2,C_2, \ldots, A_n,C_n,TC) \]

- 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 security levels.
Multilevel relation

SELECT * FROM EMPLOYEE

<table>
<thead>
<tr>
<th>Name</th>
<th>Salary</th>
<th>JobPerformance</th>
<th>TC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smith</td>
<td>U</td>
<td>40000 C</td>
<td>S</td>
</tr>
<tr>
<td>Brown</td>
<td>C</td>
<td>80000 S</td>
<td>S</td>
</tr>
</tbody>
</table>

- A user with security level S
Multilevel relation

SELECT * FROM EMPLOYEE

<table>
<thead>
<tr>
<th>Name</th>
<th>Salary</th>
<th>JobPerformance</th>
<th>TC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smith</td>
<td>U</td>
<td>40000</td>
<td>C</td>
</tr>
<tr>
<td>Brown</td>
<td>C</td>
<td>80000</td>
<td>S</td>
</tr>
</tbody>
</table>

➤ A user with security level C

<table>
<thead>
<tr>
<th>Name</th>
<th>Salary</th>
<th>JobPerformance</th>
<th>TC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smith</td>
<td>U</td>
<td>40000</td>
<td>C</td>
</tr>
<tr>
<td>Brown</td>
<td>C</td>
<td>null</td>
<td>C</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Good</td>
<td>C</td>
</tr>
</tbody>
</table>
Multilevel relation

```
SELECT * FROM EMPLOYEE
```

A user with security level U

<table>
<thead>
<tr>
<th>Name</th>
<th>Salary</th>
<th>JobPerformance</th>
<th>TC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smith U</td>
<td>40000 C</td>
<td>Fair</td>
<td>S</td>
</tr>
<tr>
<td>Brown C</td>
<td>80000 S</td>
<td>Good</td>
<td>C</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Salary</th>
<th>JobPerformance</th>
<th>TC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smith U</td>
<td>null</td>
<td>null</td>
<td>U</td>
</tr>
<tr>
<td>Brown C</td>
<td>null</td>
<td>null</td>
<td>U</td>
</tr>
</tbody>
</table>
Multilevel relation

SELECT * FROM EMPLOYEE

- A user with security level U

<table>
<thead>
<tr>
<th>EMPLOYEE</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Smith</td>
<td>U</td>
<td>40000</td>
<td>C</td>
<td>S</td>
</tr>
<tr>
<td>Brown</td>
<td>C</td>
<td>80000</td>
<td>S</td>
<td>S</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>EMPLOYEE</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Smith</td>
<td>U</td>
<td>null</td>
<td>U</td>
<td>null</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TC</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Smith</td>
<td>S</td>
<td>S</td>
<td>S</td>
<td>S</td>
</tr>
<tr>
<td>Brown</td>
<td>C</td>
<td>S</td>
<td>S</td>
<td>S</td>
</tr>
</tbody>
</table>

- A user with security level U
### Mandatory Access Control (4)

#### (a) EMPLOYEE

<table>
<thead>
<tr>
<th>Name</th>
<th>Salary</th>
<th>JobPerformance</th>
<th>TC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smith</td>
<td>U 40000</td>
<td>C Fair</td>
<td>S</td>
</tr>
<tr>
<td>Brown</td>
<td>C 80000</td>
<td>Good</td>
<td>S</td>
</tr>
</tbody>
</table>

#### (b) EMPLOYEE

<table>
<thead>
<tr>
<th>Name</th>
<th>Salary</th>
<th>JobPerformance</th>
<th>TC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smith</td>
<td>U 40000</td>
<td>null</td>
<td>C</td>
</tr>
<tr>
<td>Brown</td>
<td>C null</td>
<td>Good</td>
<td>C</td>
</tr>
</tbody>
</table>

#### (c) EMPLOYEE

<table>
<thead>
<tr>
<th>Name</th>
<th>Salary</th>
<th>JobPerformance</th>
<th>TC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smith</td>
<td>U null</td>
<td>null</td>
<td>U</td>
</tr>
</tbody>
</table>

#### (d) EMPLOYEE

<table>
<thead>
<tr>
<th>Name</th>
<th>Salary</th>
<th>JobPerformance</th>
<th>TC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smith</td>
<td>U 40000</td>
<td>C Fair</td>
<td>S</td>
</tr>
<tr>
<td>Smith</td>
<td>U 40000</td>
<td>Excellent</td>
<td>C</td>
</tr>
<tr>
<td>Brown</td>
<td>C 80000</td>
<td>Good</td>
<td>S</td>
</tr>
</tbody>
</table>
Properties of Multilevel relation

**Read and write operations:** satisfy the No Read-Up and No Write-Down principles.
Properties of Multilevel relation

**Entity integrity**: 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.
Polyinstantiation: where several tuples can have the same apparent key value but have different attribute values for users at different classification levels.
A user with security level C tries to update the value of JobPerformance of Smith to ‘Excellent’:

UPDATE EMPLOYEE
SET JobPerformance = ‘Excellent’
WHERE Name = ‘Smith’;
# Polyinstantiation example

<table>
<thead>
<tr>
<th>Name</th>
<th>Salary</th>
<th>Job Performance</th>
<th>TC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smith U</td>
<td>40000</td>
<td>C</td>
<td>S</td>
</tr>
<tr>
<td>Smith U</td>
<td>40000</td>
<td>C</td>
<td>S</td>
</tr>
<tr>
<td>Brown C</td>
<td>80000</td>
<td>S</td>
<td>C</td>
</tr>
</tbody>
</table>
Comparing DAC and 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.
Comparing DAC and MAC (2)

- 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.
Outline

Introduction to Database Security Issues
- Types of Security
- Threats to databases
- Database Security and DBA
- Access Protection, User Accounts, and Database Audits

Discretionary Access Control
- Types of Discretionary Privileges
- Specifying Privileges Using Views
- Revoking Privileges
- Propagation of Privileges Using the GRANT OPTION
- An example
- Weakness

Mandatory Access Control
- Bell-LaPudula Model
- Comparing DAC and MAC

RBAC (Role-Based Access Control)

Encryption & PKI (Public Key Infrastructure)
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 enterprisewide 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.
Outline

Introduction to Database Security Issues

• Types of Security
• Threats to databases
• Database Security and DBA
• Access Protection, User Accounts, and Database Audits

Encryption & PKI (Public Key Infrastructure)

RBAC (Role-Based Access Control)

• Comparing DAC and MAC
• Bell-LaPudula Model

Mandatory Access Control

Weaknesses

An example

Propagation of Privileges Using the GRANT OPTION

• Revoking Privileges
• Specifying Privileges Using Views

Discretionary Access Control

Access Protection, User Accounts, and Database Audits

Database Security and DBA

Types of Security

Rights to databases
Encryption

- The encoding of the data by a special algorithm that renders the data unreadable by any program without the decryption key
- Symmetric cryptography: sender and receiver use the same key
- Asymmetric cryptography: encryption & decryption keys
Encryption

- **Plaintext** is the original content which is readable as textual material. Plaintext needs protecting.

- **Ciphertext** is the result of encryption performed on plaintext using an algorithm. Ciphertext is not readable.

- **Cryptosystems** = encryption + decryption algorithms

- Encryption, decryption process needs **keys**
Encryption

- **Symmetric (shared-/secret-key) cryptosystem**: the same key for (en/de)cryption algorithms
- **Asymmetric (public-key) cryptosystem**: public & private keys
Encryption

- (Most popular) Symmetric techniques: DES, AES
  - The same key is used for both encryption and decryption
  - Faster than encryption and decryption in public-key (PK) cryptosystems
  - Less security comparing to encryption and decryption in PK cryptosystems

- Asymmetric techniques: RSA, DSA
Encryption

- **DES: Data Encryption Standard**
  - A message is divided into 64-bit blocks
  - Key: 56 bits
  - Brute-force or exhaustive key search attacks (now: some hours): see 7.6.3

- **Triple DES**: run the DES algorithm a multiple number of times using different keys
  - Encryption: \( c \leftarrow \mathcal{E}_{k_1} (\mathcal{D}_{k_2} (\mathcal{E}_{k_1} (m))) \)
  - Decryption: \( m \leftarrow \mathcal{D}_{k_1} (\mathcal{E}_{k_2} (\mathcal{D}_{k_1} (c))) \)
  - The triple DES can also use three different keys
Encryption

- AES: Advanced Encryption Standard (Rijndael)
  - Jan 2, 1997, NIST announced the initiation of a new symmetric-key block cipher algorithm, AES, as the new encryption standard to replace the DES
  - Oct 2, 2000: Rijndael was selected. Rijndael is designed by two Belgium cryptographers: Daemen and Rijmen
- Rijndael is a block cipher with a variable block size and variable key size
- The key size and the block size can be independently specified to 128, 192 or 256 bits
Cryptography-related concepts

- **RSA:** named after 3 inventors Rivest, Shamir và Adleman
  - Two keys: public key and private key
  - Public key is used for encryption.
  - Private key is used for decryption
Encryption

- Encryption key: public key
- Decryption key: private key
- Asymmetric techniques: more secure but expensive in terms of computational costs

Sender

Use public key of receiver to encrypt the message encryption key

Encrypted message using a symmetric key

Receiver
Cryptography-related concepts

- PKI (Public Key Infrastructure) and digital certificates
  - CA (certificate authority)

Alice to CA

Bob to CA
Encryption & PKI (Public Key Infrastructure)

- How does PKI work?

Certificate Authority (CA)

TRUSTED

1. Ask for R’s public key
2. Receive R’s public key
3. Send data
4. Receive data and decrypt it

Sender S

Use R’s public key to encrypt the message encryption key

Encrypted message using a symmetric key

Receiver R
Summary

Introduction to Database Security Issues
- Types of Security
- Threats to databases
- Database Security and DBA
- Access Protection, User Accounts, and Database Audits

Discretionary Access Control
- Types of Discretionary Privileges
- Specifying Privileges Using Views
- Revoking Privileges
- Propagation of Privileges Using the GRANT OPTION
- An example
- Weakness

Mandatory Access Control
- Bell-LaPadula Model
- Comparing DAC and MAC

RBAC (Role-Based Access Control)

Encryption & PKI (Public Key Infrastructure)