Information Systems Security

Lecture 9

Database Security

Dr. En. Bader Ahmad

Outline

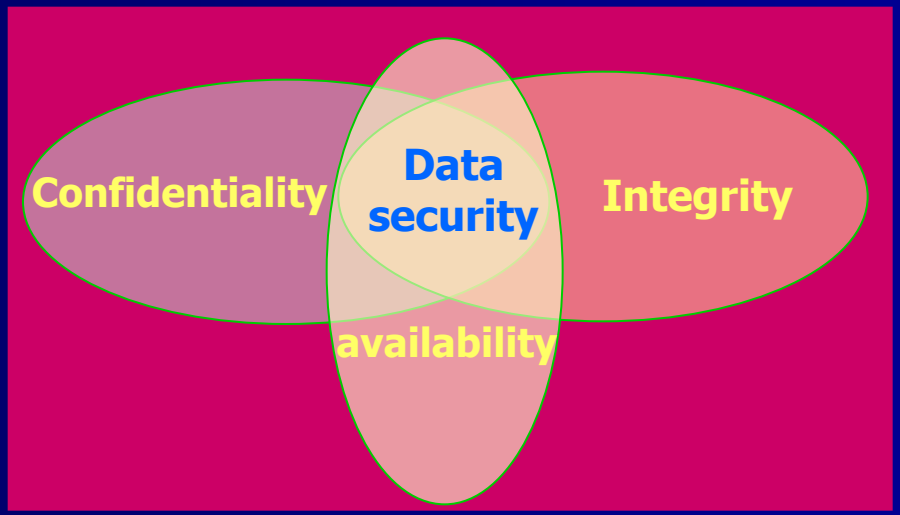
1. Data Security
2. Access control
3. Security policy
4. Access control policy Model
   - 4.1. Discretionary Access Control (DAC)
   - 4.2. Content-Based Access Control (CBAC)
   - 4.3. Mandatory Access Control (MAC)
   - 4.4. Role-Based Access Control (RBAC)

1. Data security

1. Consider a payroll database in a corporation, it must be ensured that:
   - Salaries of individual employees are not disclosed to arbitrary users of the database,
   - Salaries are modified by only those individuals that are properly authorized,
   - Paychecks are printed on time at the end of each pay period.
2. In a military environment, it is important that:
   - The target of a missile is not given to an unauthorized user,
   - The target is not arbitrarily modified,
   - The missile is launched when it is fired.

Data Security: main goals

- ***Confidentiality***: it refers to data protection from unauthorized read operations.
- ***Integrity***: it refers to data protection from unauthorized modification operations.
- ***Availability***: it ensures that data access is not denied to authorized subjects.
- Others: Authentication, etc.

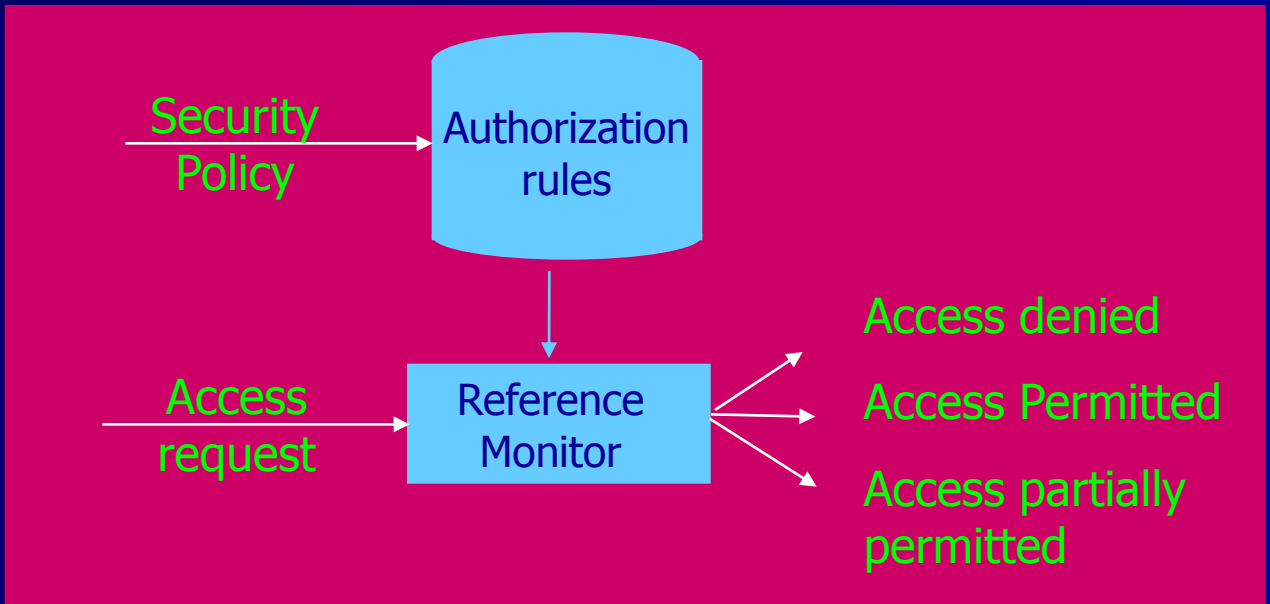


Data Security: main goals

- **Confidentiality** is enforced by the access control mechanism.
- **Integrity** is enforced by the access control mechanism and by the semantic integrity constraints specified during schema definition.
- **Availability** is enforced by the recovery, and concurrency control mechanisms, and DoS protection.

2. Access control: Basic concepts

- An **access control system** regulates the operations that can be executed on data and resources to be protected.
- Its goal is to control operations executed by subjects in order to prevent actions that could damage data and resources.



3. Security Policy

- **Policies** deal with defining *what* is authorized and *who* can grant authorizations.
- Existing security policies tend to focus mainly on the confidentiality (Privacy) requirements of security.
- Policies are used like requirements; they are the starting point in the development of any system that has security features.
- Adopted security policies mainly depend on organizational requirements, such as legal requirements, regulatory requirements, user requirements.

Security Policies and Authorizations

- The security policies are implemented by mapping them into a set of ***authorizations***
- Authorizations thus establish the operations and rights that subjects can exercise on the protected objects
- The ***reference monitor*** is a control mechanism
  - It has the task of determining whether a given subject is authorized to access the data

4. Access control policy Model

- Discretionary Access Control (DAC)
- Mandatory Access Control (MAC)
- Role-Based Access Control (RBAC)
- Context-Based Access Control (CBAC)

Access control Policy

- Most access control policies are formulated in terms of ***subjects***, ***objects***, and ***privileges***
- **Authorization Subjects**: An abstraction of any active entity that performs computation in the system
  - Examples: users, processes, roles, etc.
- **Authorization Objects**: Anything that holds data, such as relations, directories, inter-process messages, network packets, I/O devices, tables, views, or physical media
- **Authorization Privileges**: Operations that a subject can exercise on the objects in the system
  - Examples: read, write, execute, select, insert, update, delete, etc.

4.1. DAC

- **DAC** policies govern the access of subjects to objects on the basis of subjects' identity and authorization rules.
- When an access request is submitted to the system, the access control mechanism verifies whether there is an authorization rule authorizing the access.
- Such mechanisms are discretionary in that they allow subjects to grant other subjects authorization to access their objects at their discretion.
- Most of the common commercial DBMSs support it

DAC: SQL commands

- Privilege delegation is supported through the ***grant*** option:
  - if a privilege is granted with the *grant* option, the user receiving it can not only exercise the privilege, but can also grant it to other users
- A user can only grant a privilege on a given table if he/she is the table owner or if he/she has received the privilege with *grant* option
- `GRANT` *`PrivilegeList`*`| ALL[PRIVILEGES] ON` *`table`*`|`*`View`* `TO` *`UserList`* `| PUBLIC [WITH GRANT OPTION]`

DAC: SQL commands

- Example:

```
Bob: GRANT select, insert ON Employee TO
  Ann WITH GRANT OPTION;
Bob: GRANT select ON Employee TO Jim WITH
  GRANT OPTION;
Ann: GRANT select, insert ON Employee TO
  Jim;
```

  - Jim has the *select* privilege (received from both Bob and Ann) and the *insert* privilege (received from Ann).
  - Jim can *grant* to other users the select privilege (because it has received it *with grant* option); however, he cannot grant the *insert* privilege.

DAC in SQL - Grant

- Grant Command: Example

```
Bob: GRANT select, insert ON Employee TO Jim WITH
   GRANT OPTION;
Bob: GRANT select ON Employee TO Ann WITH GRANT
   OPTION;
Bob: GRANT insert ON Employee TO Ann;
Jim: GRANT update ON Employee TO Tim WITH GRANT
   OPTION;
Ann: GRANT select, insert ON Employee TO Tim;
```

- The first three GRANT commands are fully executed (Bob is the owner of the table)
- The fourth command is not executed, because Jim does not have the *update* privilege on the table
- The fifth command is partially executed; Ann has the *select* and *insert* but she does not have the *grant* option for the *insert*
  - Tim only receives the *select* privilege

DAC in SQL - Revoke

- `REVOKE` *`PrivilegeList`*`| ALL[PRIVILEGES]`
  `ON` *`table / View`* `FROM` *`UserList`* `| PUBLIC`
- A user can only revoke the privileges he/she has granted;
- Upon execution of a *revoke* operation, the user from whom the privileges have been revoked looses these privileges, unless has them from some source independent from that has executed the *revoke*.
- Recursive revocation: whenever a user revokes an authorization on a table from another user, all the authorizations that the revokee had granted because of the revoked authorization are removed.

DAC in SQL - Revoke

- Example:
  - Bob: `GRANT select ON Employee TO Jim WITH GRANT OPTION;`
  - Bob: `GRANT select ON Employee TO Ann WITH GRANT OPTION;`
  - Jim: `GRANT select ON Employee TO Tim;`
  - Ann: `GRANT select ON Employee TO Tim;`
  - Jim: `REVOKE select ON Employee FROM Tim;`

  - Tim continues to hold the *select* privilege on table Employee after the *revoke* operation, since he has independently obtained such privilege from Ann.

4.2. CBAC

- **CBAC** conditions the access to a given object to its content.
- As an example, in a RDBMS supporting CBAC it is possible to authorize a subject to access information only of those employees whose salary is not greater than 30K.
- An approach to enforce CBAC in a DBMS:
  - by defining a ***view*** which selects the objects whose content satisfies a given condition, and then granting the authorization on the view instead of on the basic objects.

CBAC: SQL Commands

- Example: suppose we want to authorize user Ann to access only the employees whose salary is lower than 20000 – steps:
  - `CREATE VIEW Vemp AS SELECT * FROM Employee WHERE Salary < 20000;`
    `GRANT Select ON Vemp TO Ann;`
- Ann:
  - `SELECT * FROM Vemp WHERE Job = 'Programmer';`
  - This is equivalent to:
  - `SELECT * FROM Employee WHERE Salary < 20000 AND Job = 'Programmer';`