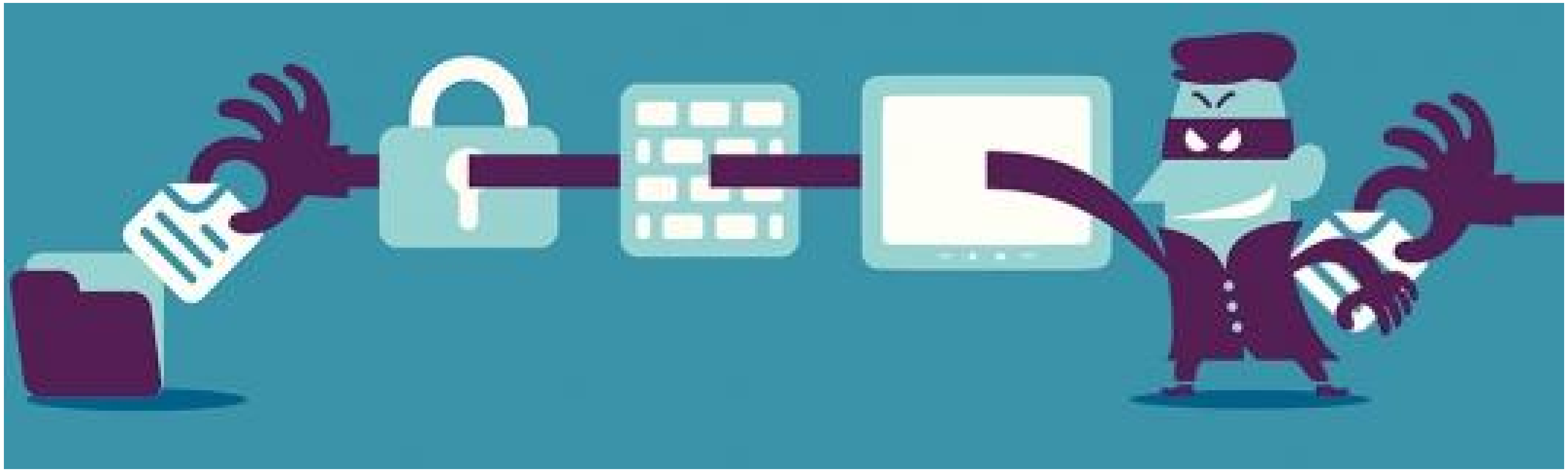
Chapter 25
Introduction to Database Security
Part 1



Chapter Outline

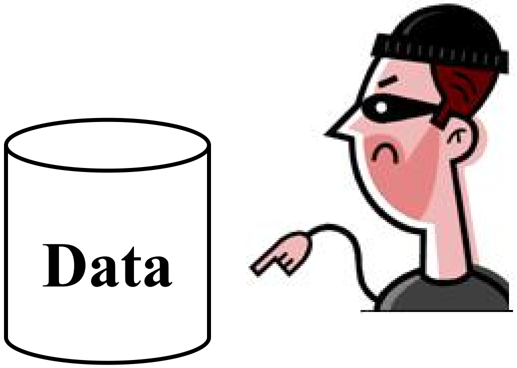
1. Introduction to Database Security Issues
2. Discretionary Access Control Based on Granting Revoking Privileges
3. Mandatory Access Control and Role-Based Access Control for Multilevel Security
4. SQL Injection
5. Introduction to Statistical Database Security
6. Introduction to Flow Control
7. Encryption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

Database Security Issues

- Legal and ethical issues
  - The right to access certain information
- Policy issues
  - At govermental, institutional, or corporate level as to what kinds of information should not be made publicly available
- System-related issues
  - Which various security functions should be enforced on systems

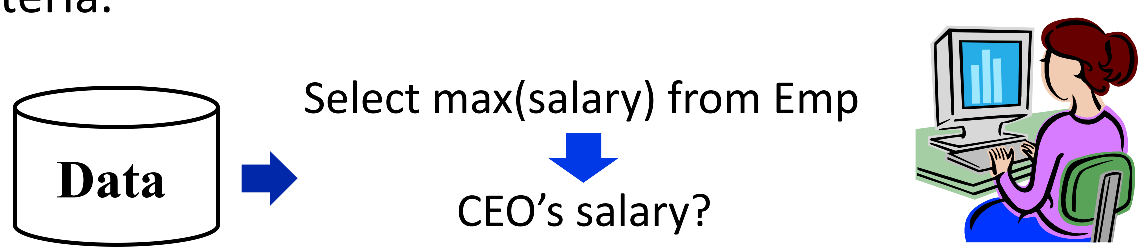
Threats and Countermeasures

- Threats to Databases
  - **Loss of integrity**
    - ✓Unauthorized changes made to the data by intentional or accidental acts
  - **Loss of availability**
    - ✓Data unavailable to users to which they have a legitimate right
  - **Loss of confidentiality**
    - ✓Refer to the protection of data from unauthorized disclosure
- Four kinds of countermeasures
  - **access control**
  - **inference control**
  - **flow control**
  - **encryption**



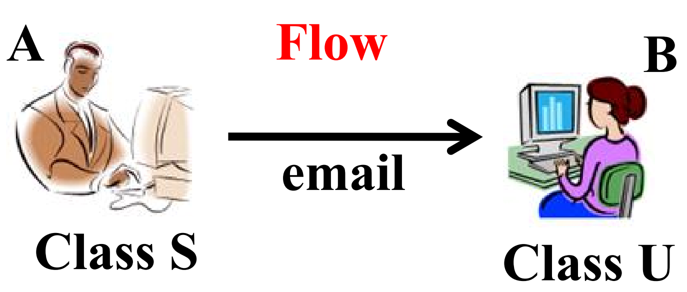
Control Measures

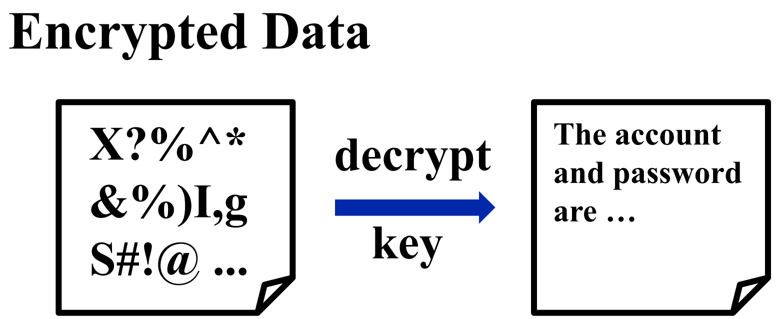
- **Access Control**
  - restricting access to the database
  - handled by using accounts and passwords to control login process by the DBMS
- **Inference Control**
  - To prevent from deducing certain facts concerning individuals from queries that involve only summary statistics on groups (or from statistical databases)
  - **Statistical database**
    - ✓used to provide statistical information or summaries of values based on various criteria.



Control Measures

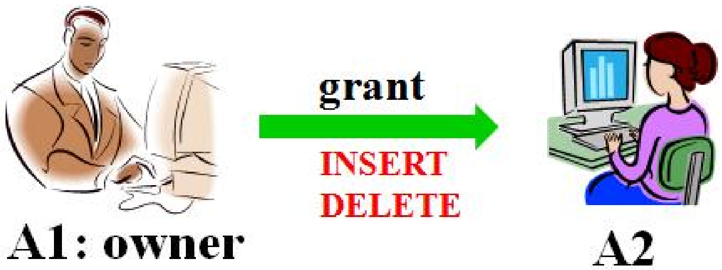
- **Flow Control**
  - To prevents information from flowing in such a way that it reaches unauthorized users.
  - Channels that are pathways for information to flow implicitly in ways that violate the security policy of an organization are called covert channels.
- **Data Encryption**
  - used to protect sensitive data (such as credit card numbers) that is being transmitted via some type communication network
  - The data is encoded using some coding 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.

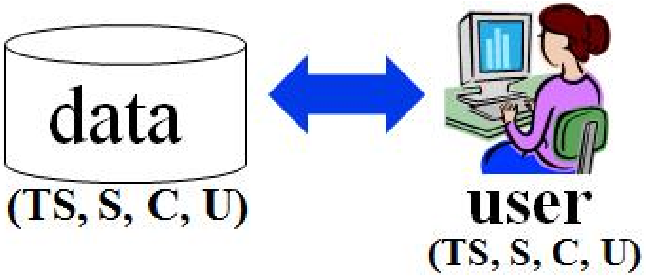




Database Security Mechanisms

- A DBMS typically includes a database security and authorization subsystem responsible for ensuring the security portions of a database against unauthorized access.
- Two types of database security mechanisms:
  - **Discretionary security mechanisms**
    - ✓To grant privileges to users, e.g. the capability to access specific data
  - **Mandatory security mechanisms**
    - ✓To enforce multilevel security by classifying the data and users into various security classes

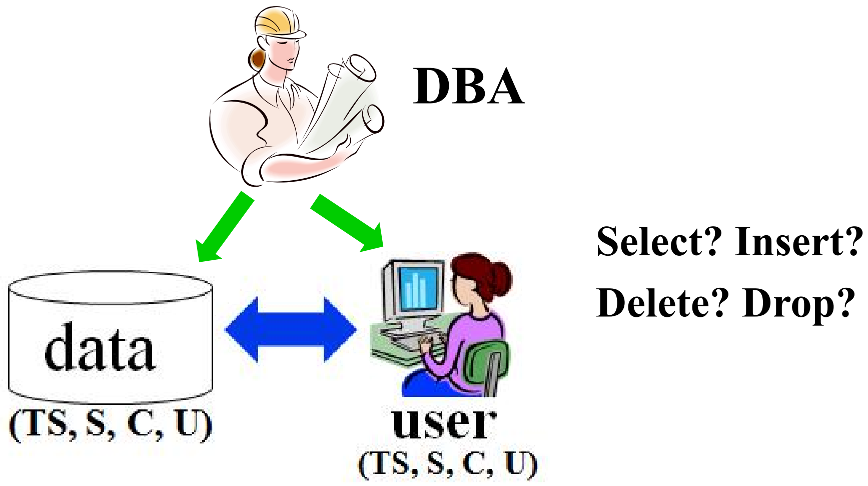




1.2 Database Security and the DBA

- The DBA is responsible for the overall security of the database system.

  1. **Account creation**
  2. **Privilege granting**
  3. **Privilege revocation**
  4. **Security level assignment**

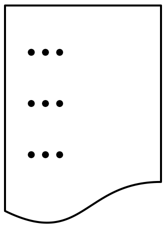


- Action 1 is **access control**, whereas 2 and 3 are **discretionary** and 4 is used to control **mandatory authorization**.

Database Audits

- DBMS must keep track of all operations 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.
- **Database Audit**
  - reviewing the log to examine all accesses and operations applied to the database during a certain time period.
- **Audit Trail**
  - A database log that is used mainly for security purposes

…
200. 15:30:20 leeys inserts employee …
201. 15:32:30 leeys deletes project …
…



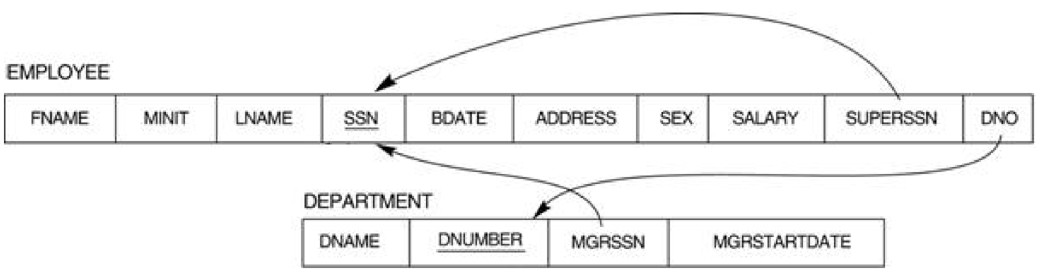
System log

Chapter Outline

1. Introduction to Database Security Issues
2. Discretionary Access Control Based on Granting 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 of Database Security
10. Oracle Label-Based Security

Discretionary Access Control Based on Granting and Revoking Privileges

- Enforcing **discretionary access control** is based on the **granting** and **revoking privileges**.
- Types of Discretionary Privileges
  - The ***account level***
    - ✓specifies the particular privileges that each account holds independently of the relations.
  - The ***relation (or table) level***
    - ✓control the privilege to access each individual relation or view.



account level

(select? update? …)

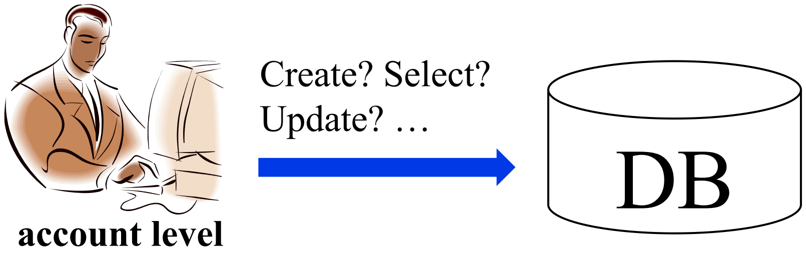
relation level

(select? update? …)

Privileges at Account Level

- The privileges at the **account level** can include
  - the **CREATE SCHEMA** or **CREATE TABLE** privilege, to create a schema or base relation;

    schema屬於架構,database(Ex,company)，table屬於表單
  - The **CREATE VIEW** privilege;
  - the **ALTER** privilege, to apply schema changes such adding or removing attributes from relations; 修改table的權利
  - the **DROP** privilege, to delete relations or views; 刪除資料表
  - the **MODIFY** privilege, to insert, delete, or update tuples;
  - the **SELECT** privilege, to retrieve information from the database by using a SELECT query.



Discretionary Privileges at Relation Level

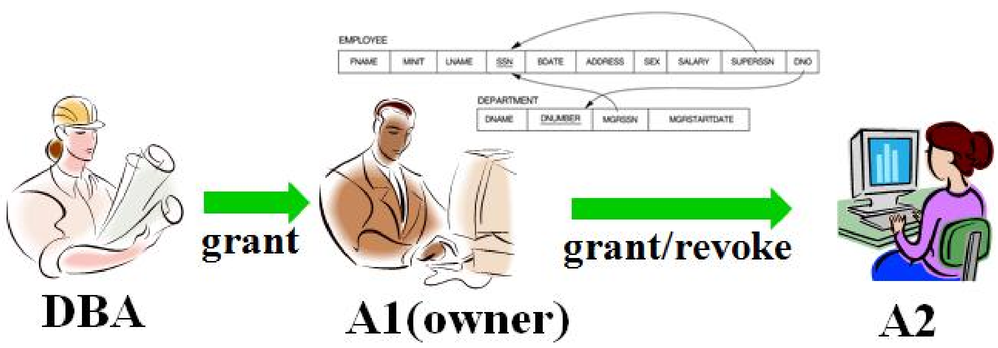
- The second level of privileges applies to the **relation level**, whether they are base relations or virtual (view) relations.
- **Access matrix model**, where the rows of a **matrix M** represents ***subjects*** (users, accounts, programs) and the columns represent ***objects*** (relations, records, 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***.

	Employee	**Project.PName**	**…**	**…**
User1	read	update	…	…
User2	read	update	…	…
Program1	update	write	…	…
…	…	…	…	…

Access Matrix

Discretionary Privileges: Grant and Revoke

- Each relation R is assigned an **owner account**, which is typically the account that was used when the relation was created.
- The **owner** is given ***all* privileges** on that relation.
- In SQL2, the DBA can assign an owner to **a whole schema** by creating the schema and associating the appropriate authorization identifier with that schema, using **CREATE SCHEMA**.
- The owner account holder can **pass privileges** on any of the owned relation to other users by **granting** privileges to their accounts.



An Example

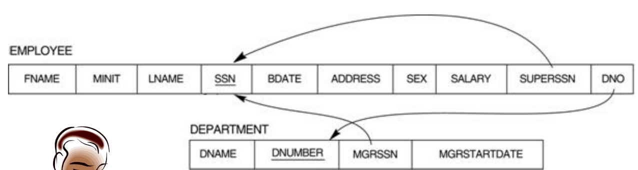
- Suppose that the DBA creates four accounts --A1, A2, A3, and A4-- and wants only A1 to be able to create base relations; then the DBA must issue the following GRANT command in SQL:

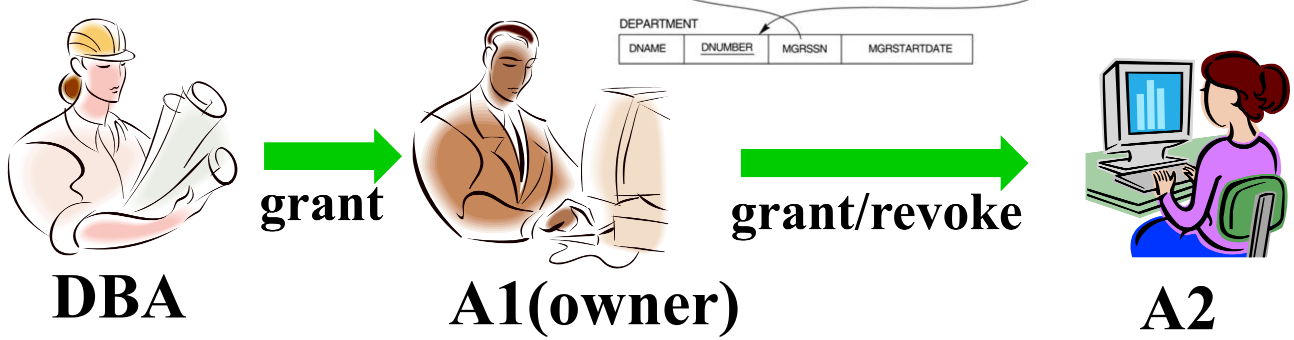
  GRANT CREATETAB TO A1;

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

  CREATE SCHEMA COMPANY AUTHORIZATION A1;

- User account A1 can create tables under the schema called **COMPANY**.

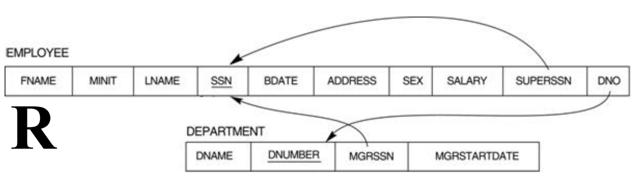




Privileges in SQL on Relation R

- **SELECT (retrieval or read) privilege on R**
  - gives the account the privilege to use the SELECT statement to retrieve tuples from R.
- **MODIFY privileges on 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.
- **REFERENCES privilege on R**
  - gives the account the capability to reference relation R when specifying integrity constraints.
  - The privilege can also be restricted to specific attributes of R.

R



An Example (2)

- **A1** creates the two base relations **EMPLOYEE** and **DEPARTMENT**;
- A1 is then **owner** and hence ***all the relation privileges*** on each of them.
- A1 wants to grant A2 the privilege to insert and delete tuples in both of these relations:

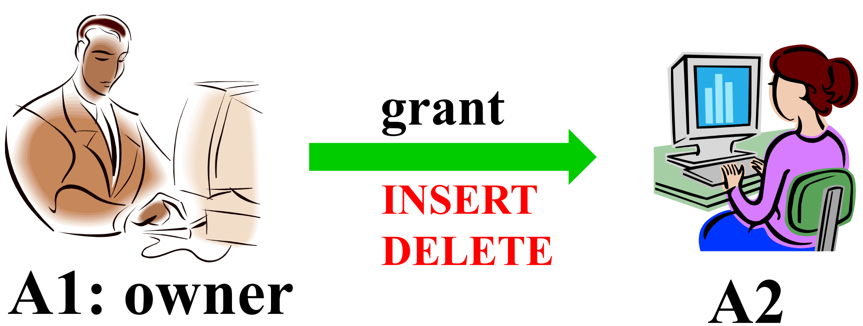
GRANT INSERT, DELETE ON EMPLOYEE, DEPARTMENT TO A2;

EMPLOYEE

NAME	SSN	BDATE	ADDRESS	SEX	SALARY	DNO

DEPARTMENT

DNUMBER	DNAME	MGRSSN



Revoking Privileges

- **REVOKE** command is for the purpose of canceling privileges.

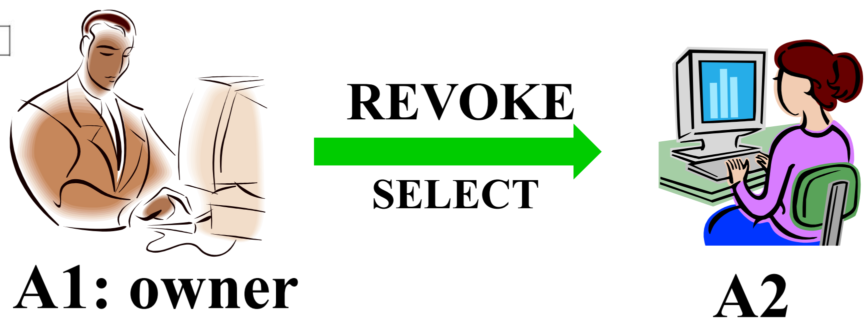
REVOKE SELECT ON EMPLOYEE FROM A2;

EMPLOYEE

NAME	SSN	BDATE	ADDRESS	SEX	SALARY	DNO

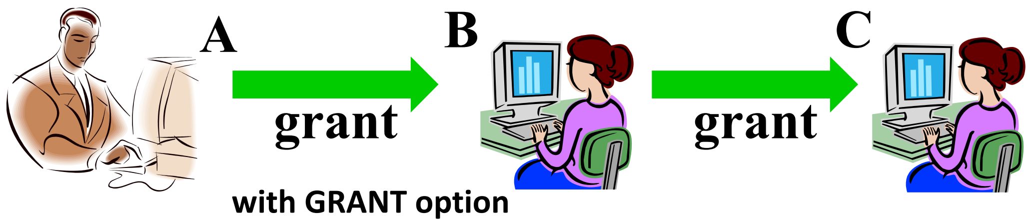
DEPARTMENT

DNUMBER	DNAME	MGRSSN



Propagation of 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
- 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.



An Example (3)

- 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;

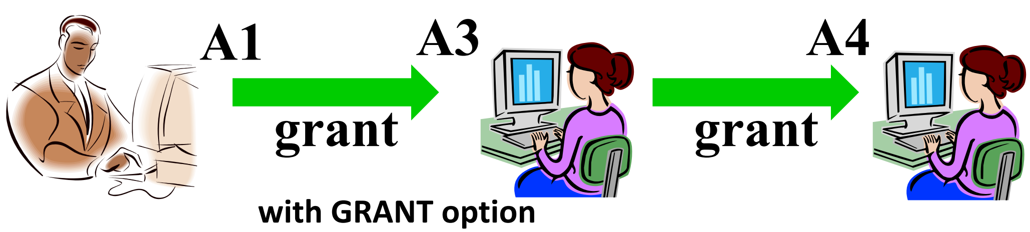
- A3 can grant the SELECT privilege on the EMPLOYEE relation to A4 by issuing:

  GRANT SELECT ON EMPLOYEE TO A4;

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

  REVOKE SELECT ON EMPLOYEE FROM A3;

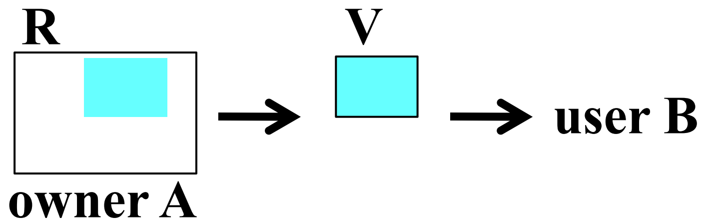
- The DBMS must automatically **revoke** the SELECT privilege on EMPLOYEE **from A4**, because A3 granted that privilege to A4 and A3 does not have the privilege any more.



2.2 Specifying Privileges Using Views

- To create a view, the account must have SELECT privilege on ***all relations*** involved in the view definition.
- The mechanism of **views** is an important discretionary authorization mechanism in its own right.

- If the owner A of a relation R wants another account B to be able to retrieve only some fields, then A can create a view V that includes only those attributes and then grant SELECT on V to B.
- The same applies to limiting B to retrieving only certain tuples; a view V' can be created by defining the view by means of a query that selects only those tuples that A wants to allow B to access.



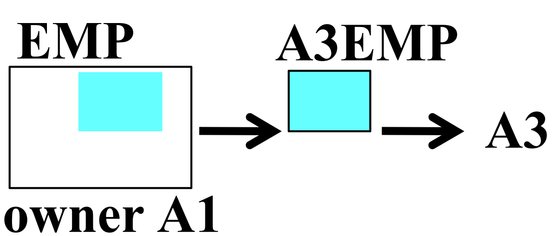
An Example(4)

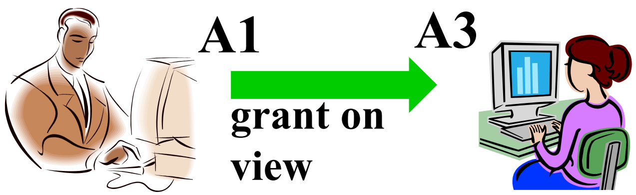
- A1 wants to give back to A3 a **limited capability** to SELECT from the EMPLOYEE relation and wants to allow A3 to be able to propagate the privilege.
- The limitation is to retrieve only the NAME, BDATE, and ADDRESS attributes and only for the tuples with DNO=5.
- A1 then create the view:

```
CREATE VIEW A3EMPLOYEE AS
SELECT NAME, BDATE, ADDRESS
FROM EMPLOYEE
WHERE DNO = 5;
```

- A1 can grant SELECT on the view A3EMPLOYEE to A3:

GRANT SELECT ON A3EMPLOYEE TO A3 WITH GRANT OPTION;





An Example (5)

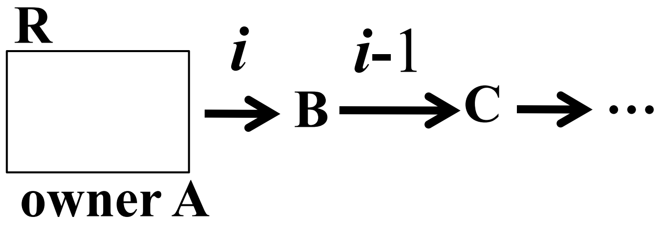
- Suppose that **A1** wants to allow **A4** to update only the SALARY attribute of EMPLOYEE;
- A1 can issue:

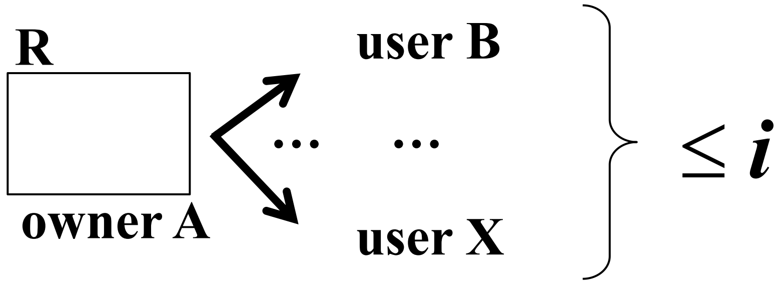
GRANT UPDATE ON EMPLOYEE (SALARY) TO A4;

- 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.

2.6 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 of privileges.





Chapter Outline

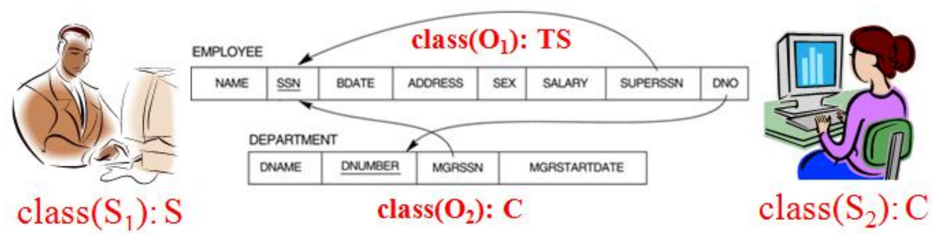
1. **Introduction to Database Security Issues**
2. **Discretionary Access Control Based on Granting Revoking Privileges**
3. **Mandatory Access Control and Role-Based Access Control for Multilevel Security**
4. **SQL Injection**
5. **Introduction to Statistical Database Security**
6. **Introduction to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

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

- The discretionary access control 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.

	Employee	**Project.PName**	**…**	**…**
User1	read	update	…	…
User2	read	update	…	…
Program1	update	insert	…	…
…	…	…	…	…

Discretionary Access Control



Mandatory Access Control

Multilevel Security: Bell-LaPadula Model

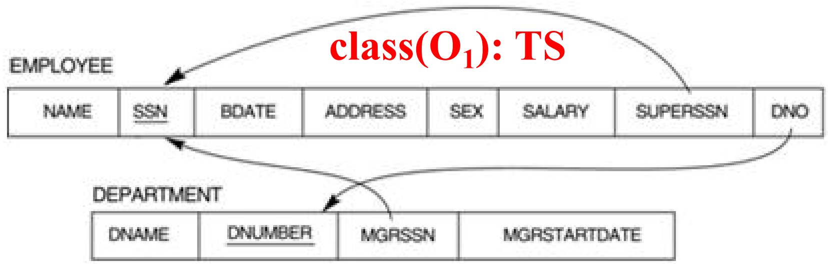
- Typical **security classes** are

  top secret (**TS**), **secret** (**S**), **confidential** (**C**), and **unclassified** (**U**), where TS is the highest level and U the lowest: **TS ≥ S ≥ C ≥ U**

- **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.

  1. 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**.
  2. A subject S is not allowed to write an object O unless class(S) ≤ class(O). This known as the **star property** (or * property).

class(S_1): S



class(O_2): C

class(S_2): C

Multilevel Security for A Relation

- Each attribute A is associated with a **classification attribute** C.
- In some models, a **tuple classification** attribute TC is added.
- 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.
- A **multilevel relation** schema R

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

  where each C_i represents the classification attribute associated with attribute A_i.
- The apparent key of a **multilevel relation** is the set of attributes that would have formed the primary key in a regular (single-level) relation.

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

Filtering

- A multilevel relation will appear to contain different data to subjects (users) with different clearance levels.
  - It is possible to store a single tuple at a higher classification level and produce the corresponding tuples at a lower-level classification through a process known as **filtering**.

The original EMPLOYEE tuples

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

Classification C users see

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

Classification U users see

Name	Salary	JobPerformance	TC
Smith U	NULL U	NULL U	U

Polyinstantiation

- Sometimes, 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.

The original EMPLOYEE tuples

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

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

Polyinstantiation of the Smith tuple

Entity Integrity Constraint in Multilevel Relation

- The **entity integrity** rule for multilevel relations states that
  1. All attributes that are members of the apparent key must not be null and must have the ***same*** security classification within each individual tuple.
  2. All other attribute values in the tuple must have a security classification **greater than or equal** to that of the apparent key.

$$R(\underline{A_1, C_1, A_2, C_2}, A_3, C_3, \ldots, A_n, C_n, TC)$$

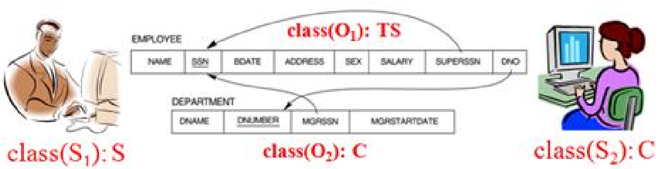
$$\text{where } C_1 = C_2 \leq C_i,\ \ 3 \leq i \leq n$$

3.1 Comparing Discretionary Access Control and Mandatory Access Control

- **Discretionary Access Control (DAC) policies**
  - **Advantage**: high degree of flexibility, which makes them suitable for a large variety of application domains.
  - **Disadvantage**: their vulnerability to malicious attacks, such as Trojan horses embedded in application programs.
- **Mandatory Access Control (MAC) policies**
  - **Advantage**: high degree of protection, which prevents any illegal flow of information.
  - **Disadvantage**: being too rigid; only applicable in limited environments.
- In many practical situations, discretionary policies are preferred because they offer a better trade-off between security and applicability.

	Employee	Project.PName	...	...
User1	read	update	...	...
User2	read	update	...	...
Program1	update	insert	...	...
...	...	...	...	...

Discretionary Access Control



Mandatory Access Control

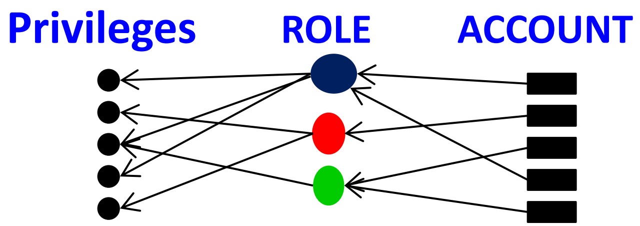
3.2 Role-Based Access Control

- **Role-based access control (RBAC)** emerged rapidly in the 1990s as a proven technology for managing and enforcing security in large-scale enterprisewide systems.
- Privileges are associated with **roles**, and users are assigned to appropriate roles,
  - such as ***sales account*, *manager*, *purchasing agent*, *department manager***, and so on.
- Roles can be created using the CREATE ROLE and DESTROY ROLE commands.
- The GRANT and REVOKE commands can then be used to assign and revoke privileges from roles.

CREATE ROLE manager;

GRANT ROLE manager TO hsucc;

GRANT ROLE sales-account TO leeys;



Role Hierarchy

- **Role hierarchy** is a natural way of organizing roles to reflect the organization’s lines of authority and responsibility.
- If a user has one role, the user automatically has roles lower in the hierarchy.

executive

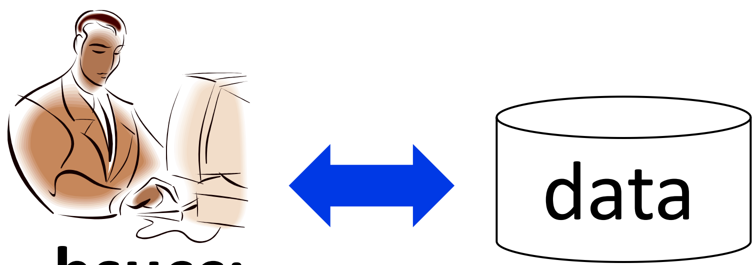
↓

manager

↓

employee

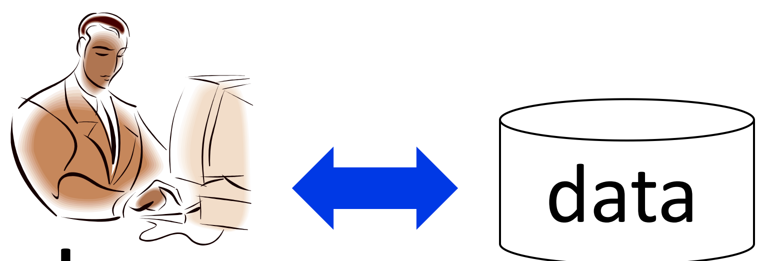
Role Hierarchy



hsucc: manager

Temporal Constraints and Web-base AP

- Another important consideration in RBAC 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.
- Discretionary access control (DAC) and mandatory access control (MAC) models **lack capabilities** needed to support the security requirements of emerging enterprises and Web-based applications.



hsucc: manager

(2010/1/1~2012/12/31)

Label-Based Security and Row-Level Access Control

- **Row-level access control**
  - Access control can be implemented by considering the data row by row.
  - Each data row is given a label, storing information about data sensitivity.
  - A user having a low authorization level is denied access to data having a higher-level number.
- Suppose a user has SELECT privileges. If the user has a sensitivity of 20, then the user can view all rows having a security level of 20 or lower.

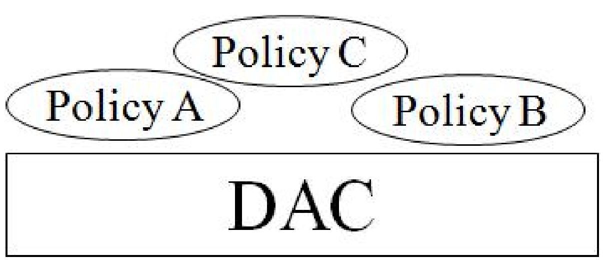
Level: 20

PROJECT

PNAME	PNUMBER	PLOCATION	DNUM	Level
ProductX	1	Bellaire	5	20
ProductY	2	Sugarland	5	15
ProductZ	3	Houston	5	25
Computerization	10	Stafford	4	21
Reorganization	20	Houston	1	10
Newbenefits	30	Stafford	4	30

Label-Based Security

- **A policy** defined by an administrator is called a **Label Security policy**.
- When a policy is implemented, **a new column** is added to each row. The added column contains the label for each row that reflects the sensitivity of the row.
- The Label Security requirements are applied on **top of the DAC** requirements.
- The user must satisfy **the DAC requirements** and then **the label security requirements** to access a row.

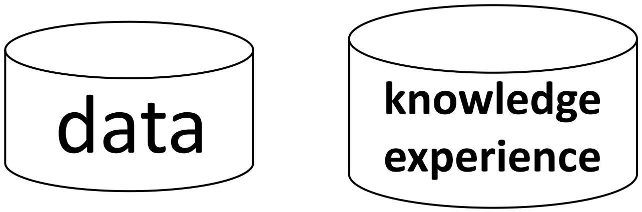


PROJECT

PNAME	PNUMBER	PLOCATION	DNUM	Levels A	Levels B	Levels C
ProductX	1	Bellaire	5			
ProductY	2	Sugarland	5			
ProductZ	3	Houston	5			
Computerization	10	Stafford	4			
Reorganization	20	Houston	1			
Newbenefits	30	Stafford	4			

3.3 Access Control Policies for E-Commerce and the Web

- **E-Commerce** environments require elaborate policies that go beyond traditional DBMSs.
  - The resources to be protected are not only traditional data but also knowledge and experience.
  - The access control mechanism should be flexible enough to support a wide spectrum of heterogeneous protection objects.



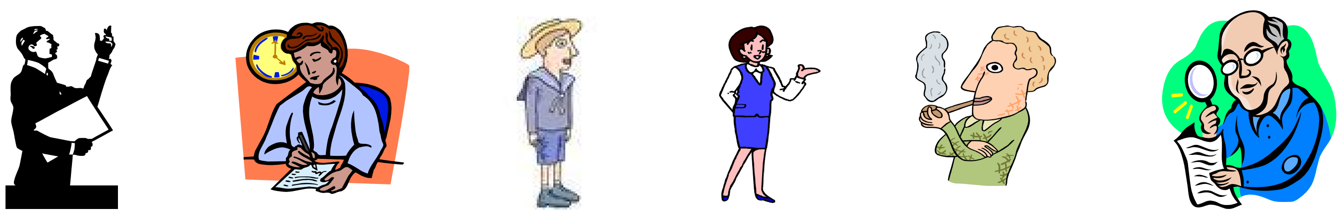
Various types of objects to be protected

(heterogeneous objects)

Access Control Policies for E-Commerce and the Web

- Another requirement is related to the heterogeneity of subjects, which requires access control policies based on user characteristics and qualifications.
  - A ***credential*** is a set of properties concerning a user that are relevant for security purposes (e.g., age, position, or role within an organization).
  - By using credentials, one can formulate policies such as

> ***Policy A:*** *Only **permanent staff** with **five or more years** of service can **access** documents related to the **internals of the system**.*



60
CEO
manager

Heterogeneous subjects

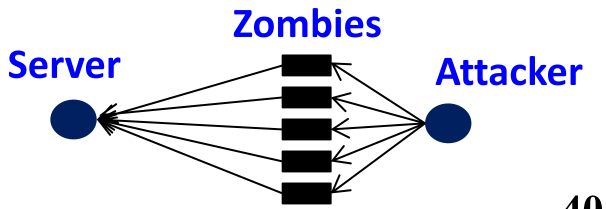
65
teacher
customer

Frequent Attacks on Databases

- **SQL Injection**
- **Unauthorized Privilege Escalation**
  - An individual attempting to elevate his privilege by attacking vulnerable points in DBMS
- **Denial of Service (DOS) Attack**
  - To make resources unavailable to its 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 their login credentials

Passwords:

abcde, 12345678, tel#, birthdate, qwerty, …



SQL Injection Methods

- Attacker injects a string input thru the application, which changes or manipulate the SQL statement to the attacker's advantage.
  - SQL Manipulation
  - Code Injection
  - Function Call Injection

SQL Manipulation

- A manipulation attack changes an SQL command in the application

SELECT *

FROM users

WHERE username = ‘jake’ and PASSWORD = ‘jakepasswd’

- The attacker can use the following:

SELECT *

FROM users

WHERE username = ‘jake’ and PASSWORD = ‘jakepasswd’ or ‘x’ = ‘x’

Code Injection

- Code injection 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.
- Code injection is a popular technique for system hacking or cracking to gain information.

statement = "SELECT * FROM users WHERE name = ' "+ userName +" ';"

The hacker input the following for the variable **userName**:

a';DROP TABLE users; SELECT * FROM userinfo WHERE 't' = 't

The SQL statement becomes:

SELECT * FROM users WHERE name = '**a';DROP TABLE users; SELECT * FROM userinfo WHERE 't' = 't';**

Code Injection

statement = "SELECT * FROM users WHERE name = ' "+ userName +" ';"

If string **userName** is: **' or '1' = '1**

SELECT * FROM users WHERE name = ' ' OR '1' = '1';

If string **userName** is: **' or '1' = '1' /* '**

SELECT * FROM users WHERE name = ' ' OR '1' = '1' /* ';

Function Call Injection

- 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
- 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.

```
SELECT SYSDATE FROM dual;

SELECT TRANSLATE ('user input', 'from_string', 'to_string') FROM dual;

SELECT TRANSLATE ("||UTL_HTTP.REQUEST ('http://129.107.2.1/')||",
        '98765432', '9876') FROM dual;
```

Function Call Injection

SELECT TRANSLATE (‘user input’, ‘from_string’, ‘to_string’) FROM dual;

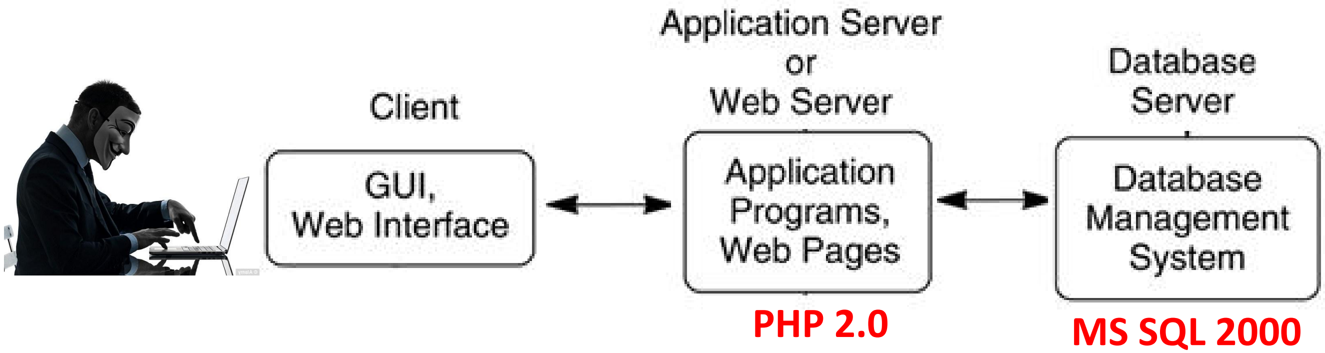
- TRANSLATE is used to replace a string of characters with another string of characters
- This type of SQL statement is subjected to a function injection attack:

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

- The attacker can 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).

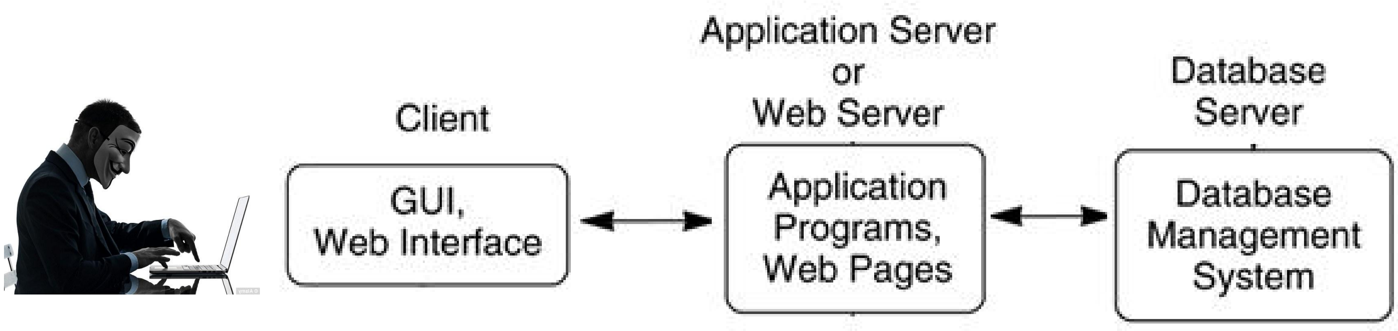
Risks Associated with SQL Injection (1)

- Database Fingerprinting
  - 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 DMBS
- Denial of Service (DOS)
- Bypassing Authentication
- Identifying Injectable Parameters
  - The attacker gathers information about the type and structure of the back-end DB 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



Risks Associated with SQL Injection (2)

- Executing Remote Commands
  - This provide attackers with a tool to execute arbitrary commands on the DB.
- Performing Privilege Escalation
  - This type of attack takes advantage of logical flaws within the DB to upgrade the access level.



Protection Techniques against SQL Injection

- **Bind Variables (Using Parameterized Statements)**

```
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)**
  - To remove escape characters from input strings by using the SQL Replace function, e.g., the single quote (‘) can be replaced by two single quotes (‘’).
- **Function Security**
  - DB functions should be restricted, as they can be exploited in the SQL function injection attacks.

```
statement = "SELECT * FROM users WHERE name = ' "+ userName +" ';"

SELECT * FROM users WHERE name = 'a';DROP TABLE users; SELECT * FROM userinfo WHERE 't' = 't';
```

Filter Input (Input Validation)

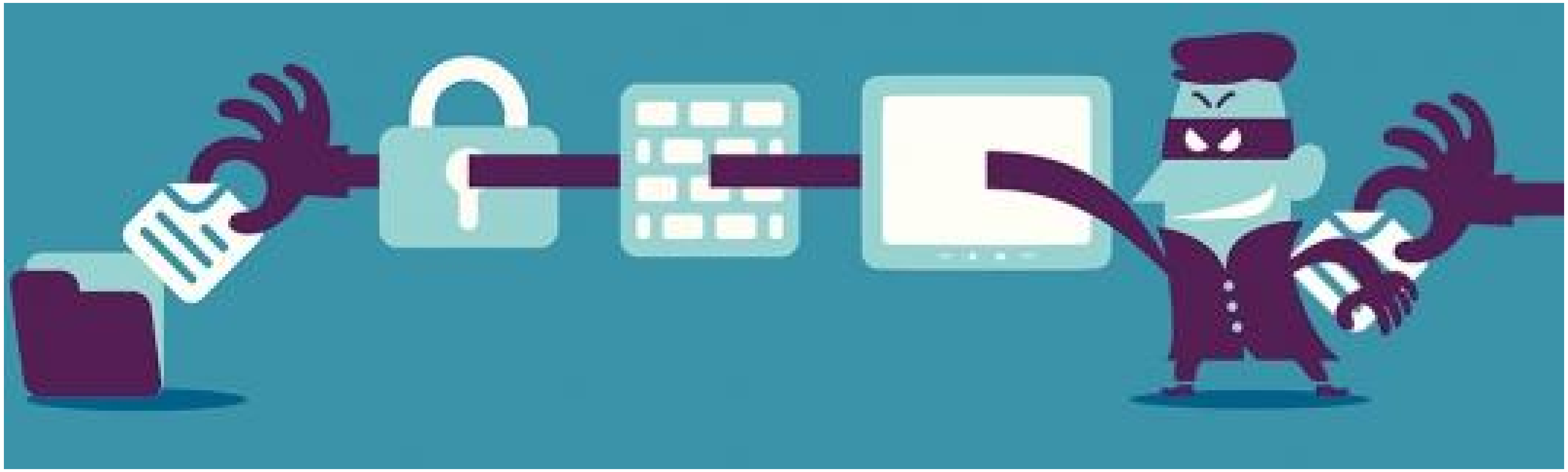
- Function **mysql_real_escape_string(String)** prepends **backslashes** (\) to the following characters in the **String:**
  \x00, \n, \r, \, ', " and **\x1a**.

- Example
  $Username: **' or '1' = '1**
  mysql_real_escape_string($Username**): \' or \'1\' = \'1**

```
$query = sprintf("SELECT * FROM Users WHERE UserName='%s' AND Password='%s'",
                    mysql_real_escape_string($Username),
                    mysql_real_escape_string($Password));           /* PHP function

mysql_query($query);
```

Chapter 25
Introduction to Database Security
Part 2

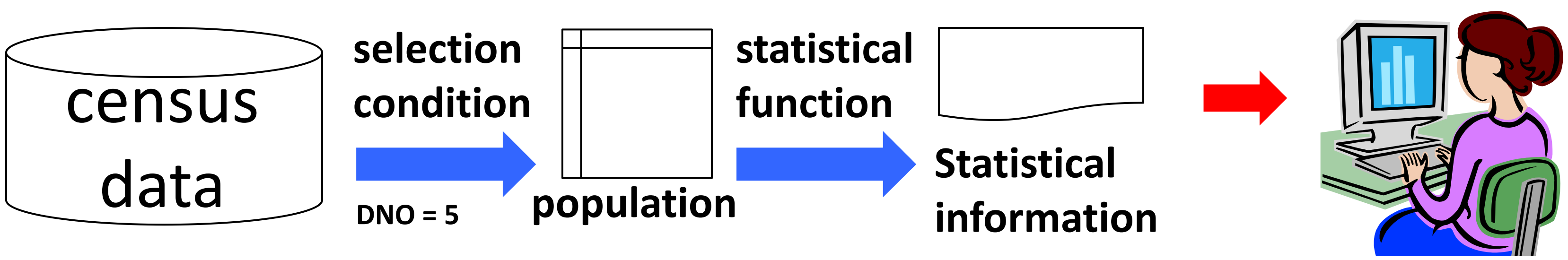


Chapter Outline

1. Introduction to Database Security Issues
2. Discretionary Access Control Based on Granting 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 of Database Security**
10. **Oracle Label-Based Security**

Statistical Database Security

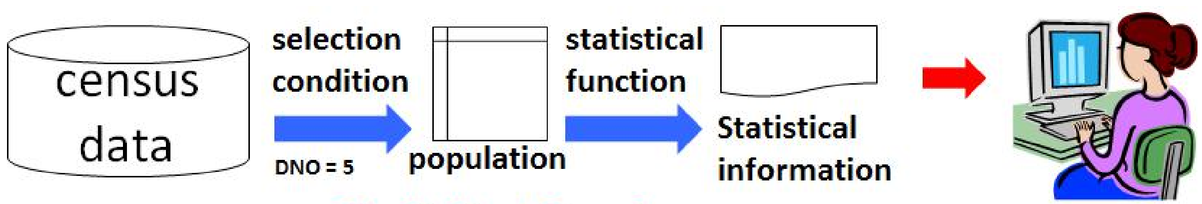
- Statistical databases are used mainly to produce statistics on various populations.
- A **population** is a set of tuples of a relation that satisfy some selection condition.
- Users are permitted to retrieve statistical information on the populations, such as averages, sums, counts, maximums, minimums, and standard deviations.
- Confidential data on individuals should be protected from user access.



Statistical Queries

Statistical Queries

- **Statistical queries** involve applying statistical functions to a population of tuples, such as
  - to retrieve the number of individuals in a population or the average income in the population.
  - not allowed to retrieve individual data, such as the income of a specific person.
- Prohibit the retrieval of individual data.
  - 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.



Statistical Queries

Inferring Individual’s Information

- It is possible to **infer** the values of individual tuples from a sequence statistical queries, especially when the conditions result in a population consisting of a small number of tuples.

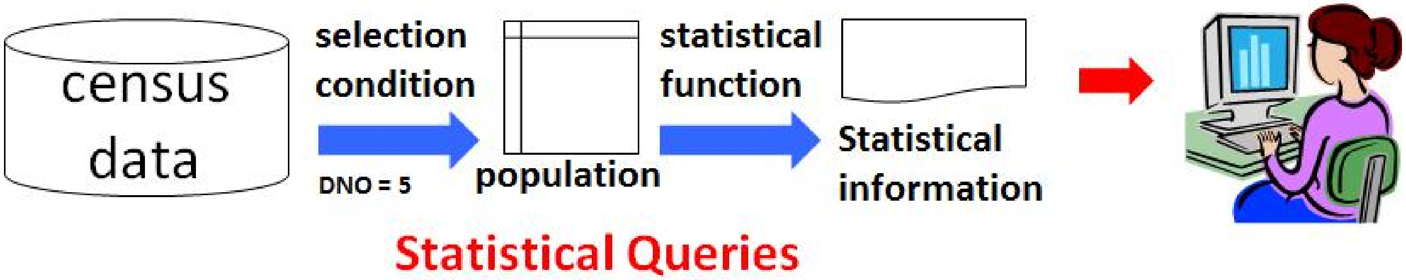
Q1: SELECT COUNT(*) FROM PERSON WHERE <condition>;

Q2: SELECT AVG(Income) FROM PERSON WHERE <condition>

- Interested in finding the Salary of Jane Smith, and we know her with (Last_degree=‘Ph.D.’ AND Sex=‘F’ AND City=‘Bellaire’ AND State=‘Texas’)
- If count ≤ 3, data can be easily inferred by using MIN, MAX, and AVG

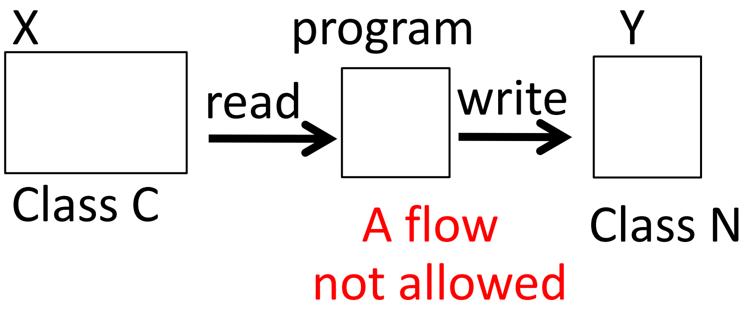
Solutions to Information Inferring

- No statistical queries are permitted whenever the number of tuples in the population specified by the selection condition falls below some threshold
- To prohibit sequences of queries that refer repeatedly to the same population of tuples
- To introduce slight inaccuracies or noise into the results of statistical queries deliberately
- To partitioning of the database; Queries can refer to any complete group(s), but never to subsets of records within a group



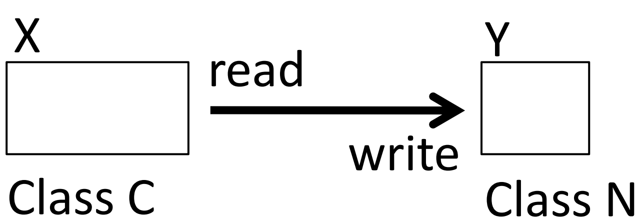
Introduction to Flow Control

- 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: confidential (C) and nonconfidential (N), and allows all flows except those from class C to class N.

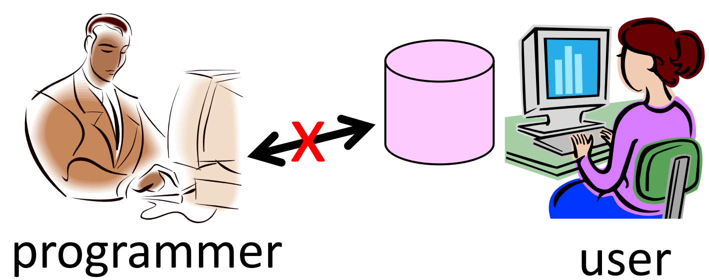


Covert Channels

- A **covert channel** allows information to pass from a higher classification level to a lower classification level through improper means.
- 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.

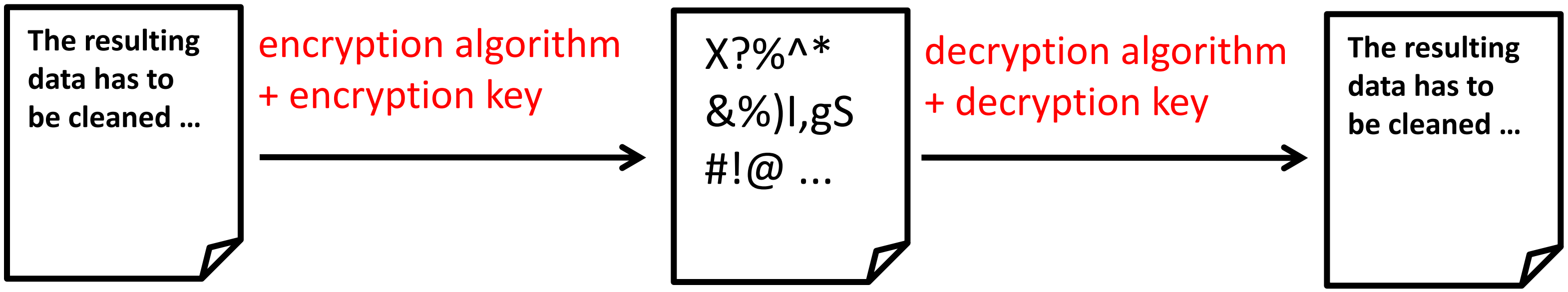


A covert channel



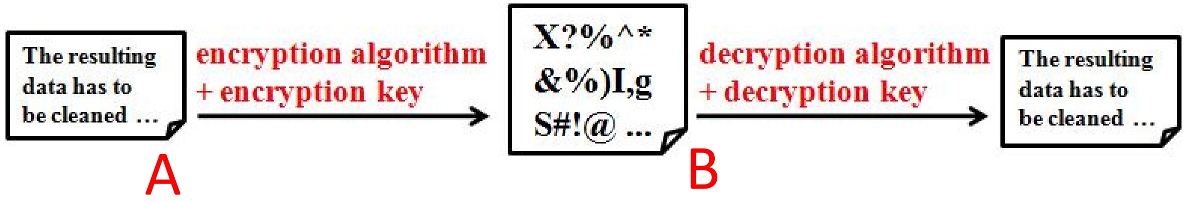
6 Encryption and Public Key Infrastructures

- **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.



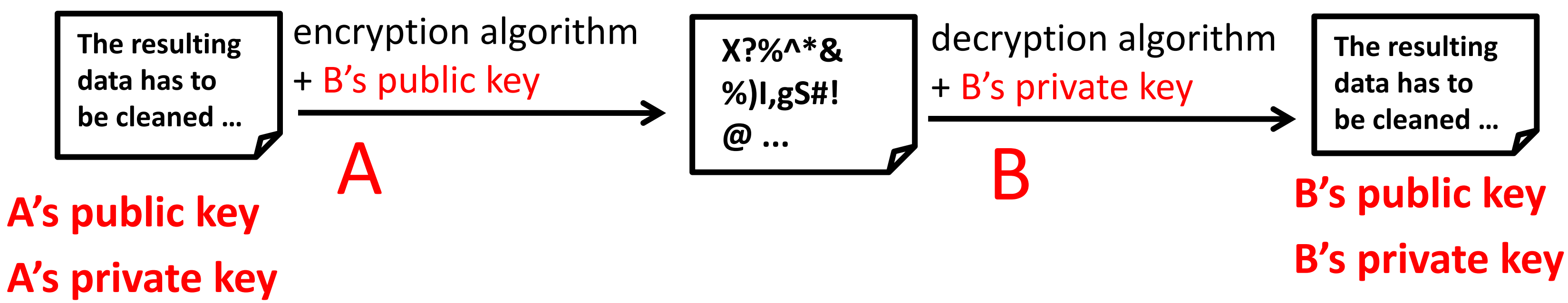
Data and Advanced Encryption Standards

- **Data Encryption Standard (DES)**
  - a system developed by the U.S. government for use by the general public.
  - widely accepted as a cryptographic standard worldwide.
  - provide end-to-end encryption on the channel between the sender A and receiver B.
- **DES** algorithm is a complex combination of two of the building blocks of encryption: **substitution** and **permutation** (transposition).
  - Its strength comes from repeated application of these two techniques for a total of 16 cycles.
  - Plaintext (the original form of the message) is encrypted as blocks of 64 bits.
- After questioning the adequacy of DES, the National Institute of Standards (NIST) introduced the **Advanced Encryption Standards (AES)**.
  - has a block size of 128 bits and thus takes longer time to crack.



Public Key Encryption

- In 1976 Diffie and Hellman proposed **public key encryption**.
- Public key algorithms are based on mathematical functions rather than operations on bit patterns.
  - Involve the use of two separate keys, in contrast to conventional encryption, which uses only one key.
- The two keys used for public key encryption are referred to as the **public key** and the **private key**.
  - The public key is made for public.
  - The private key is kept secret by owner.

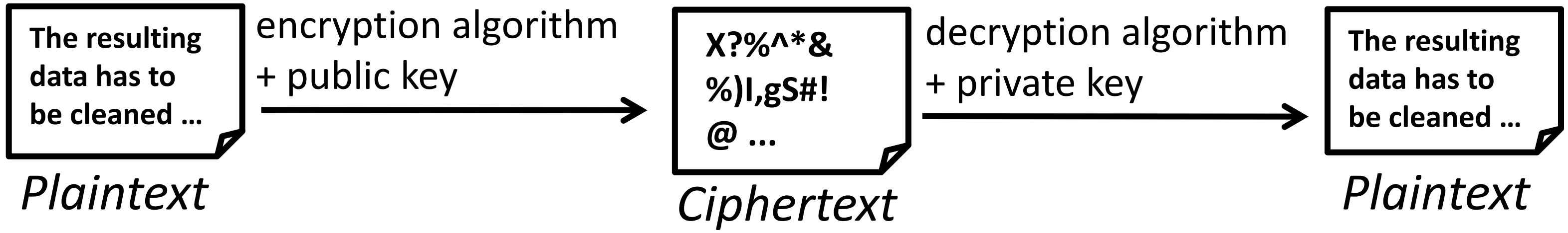


Ingredients of Public Key Encryption

1. ***Plaintext*** : the data or readable message that is fed into the algorithm as input.
2. ***Encryption algorithm*** : The algorithm performs various transformations on the plaintext.
3. ***Public and private keys*** : pair of keys that have been selected so that if one is used for encryption, the other is used for decryption.

   The exec transformations performed by the encryption algorithm depend on the public or private key that is provided as input.
4. ***Ciphertext*** : the scrambled message produced as output. It depends on the plaintext and the key.

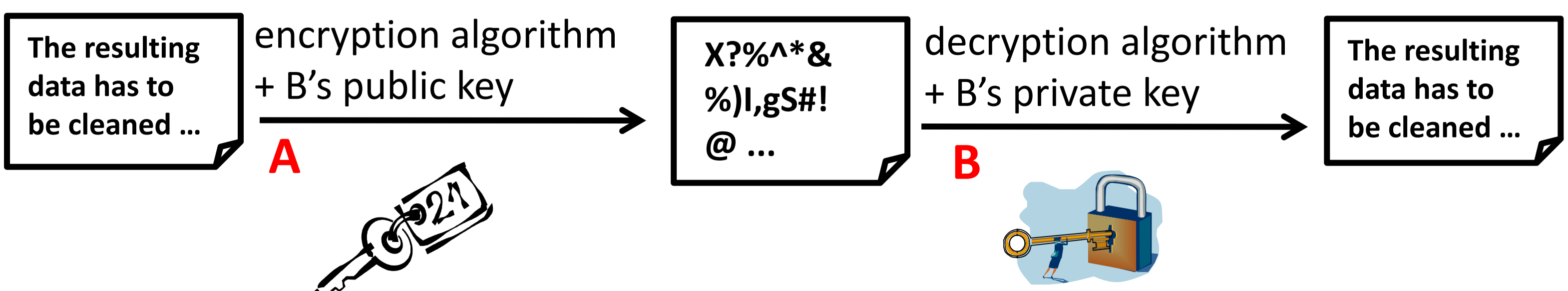
   For a given message, two different keys will produce two different ciphertexts.
5. ***Decryption algorithm*** : This algorithm accepts the ciphertext and the matching key and produces the original plaintext.



Public Key Encryption

The essential steps of a public key cryptographic algorithm:

1. Each user generates a pair of keys to be used for the encryption and decryption of messages.
2. Each user places one of the two keys in a public register or other accessible file. This is the public key. The companion key is kept private.
3. If a sender wishes to send a private message to a receiver, the sender encrypts the message using the receiver’s public key.
4. When the receiver receives the message, he or she decrypts it using the receiver’s private key. No other recipient can decrypt the message because only the receiver knows his or her private key.



Receiver’s public key   Receiver’s private key

RSA Public Key Encryption

- The RSA Public Key Encryption Algorithm
  - one of the first public key schemes was introduced in 1978 by Ron **R**ivest, Adi **S**hamir, and Len **A**dleman at MIT.
- The RSA algorithm
  - incorporates results from number theory, combined with the difficulty of determining the prime factors of a target.
  - also operates with modular arithmetic – **mod *n***.

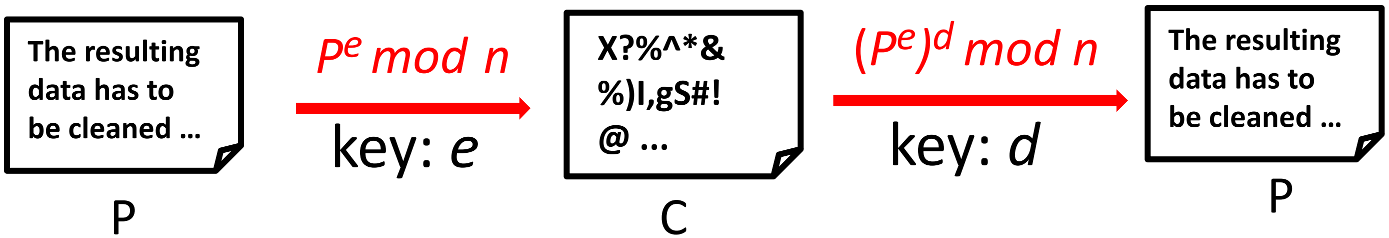
$n = a * b$ // *a, b* large prime number

$P^e \bmod n$ // encrypt with *e*

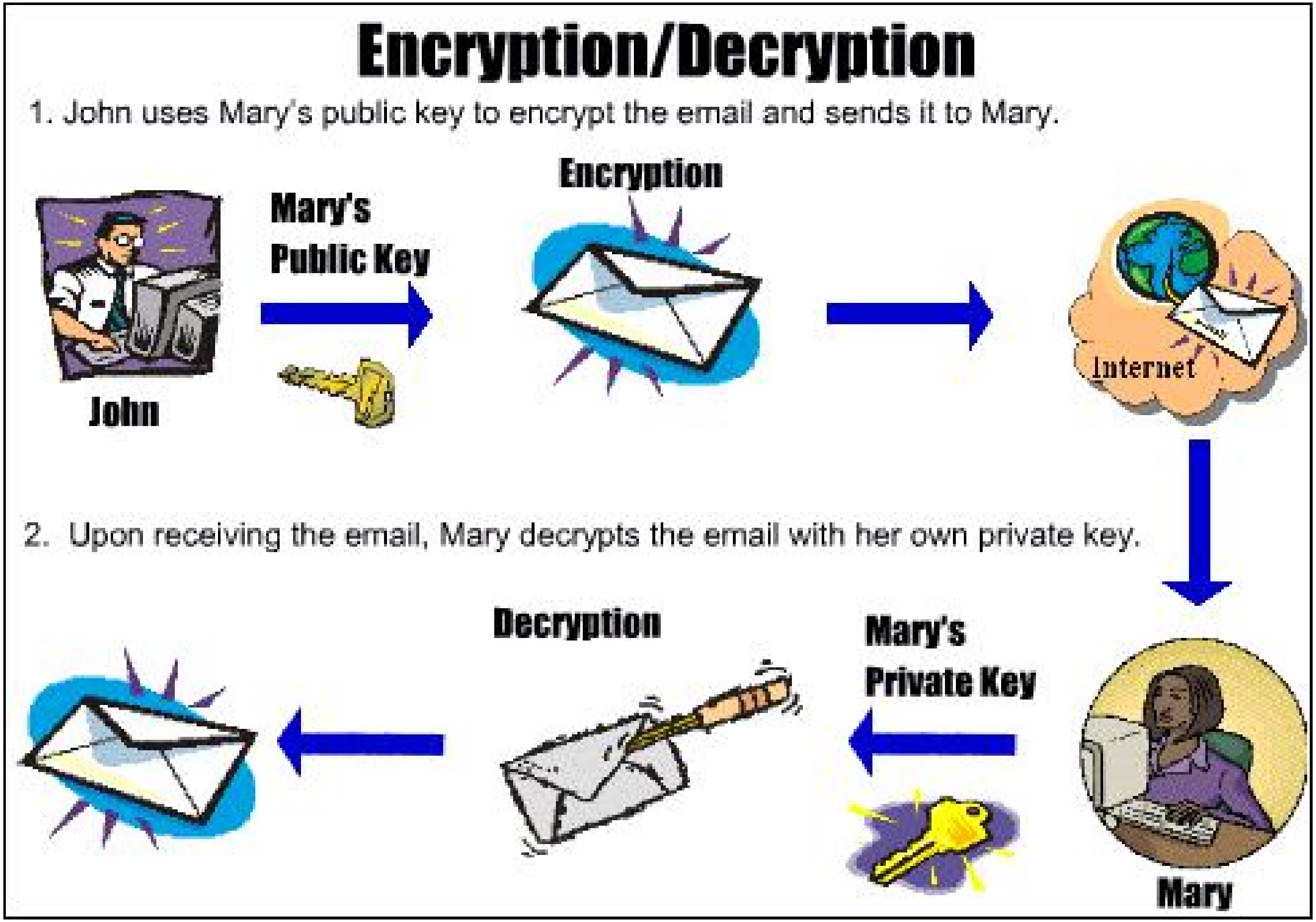
$(P^e)^d \bmod n = P$ // decrypt with *d*

RSA Public Key Encryption

- Two keys, ***d*** and ***e***, are used for *decryption* and *encryption*.
  - An important property is that d and e can be interchanged.
  - n is chosen as a large integer that is a product of two large distinct prime numbers, a and b. (i.e., $n = a \times b$)
  - The encryption key e is a randomly chosen number between 1 and n that is relatively prime to $(a\text{-}1) \times (b\text{-}1)$.
  - The plaintext block P is encrypted as $P^e \bmod n$.
  - Because the exponentiation is performed *mod n*, factoring P^e to uncover the encrypted plaintext is difficult.
  - However, the decryption key d is carefully chosen so that $(P^e)^d \bmod n = P$.
  - The decryption key d can be computed from the condition that $d \times e \equiv 1 \pmod{(a\text{-}1) \times (b\text{-}1)}$. (i.e., $(d \times e) \bmod ((a\text{-}1) \times (b\text{-}1)) = 1$)
  - Thus, the legitimate receiver who knows d simply computes $(P^e)^d \bmod n = P$ and recovers P without having to factor P^e.



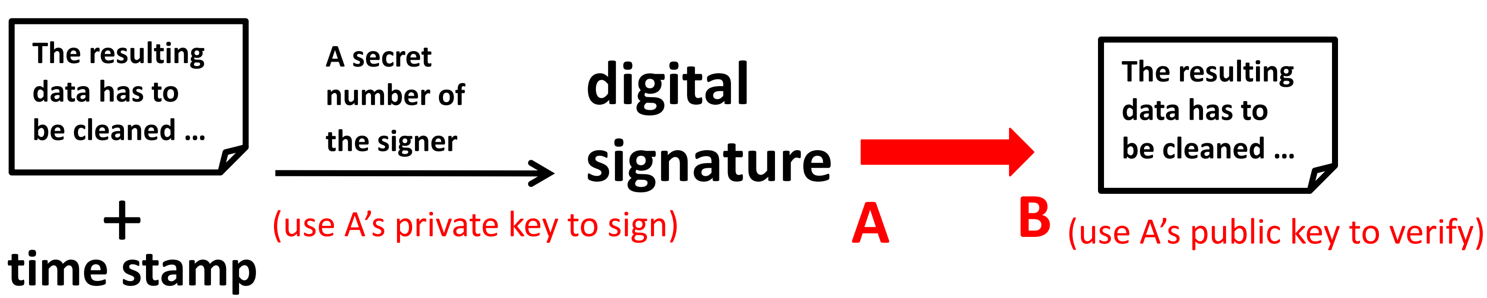
Encryption and Decryption



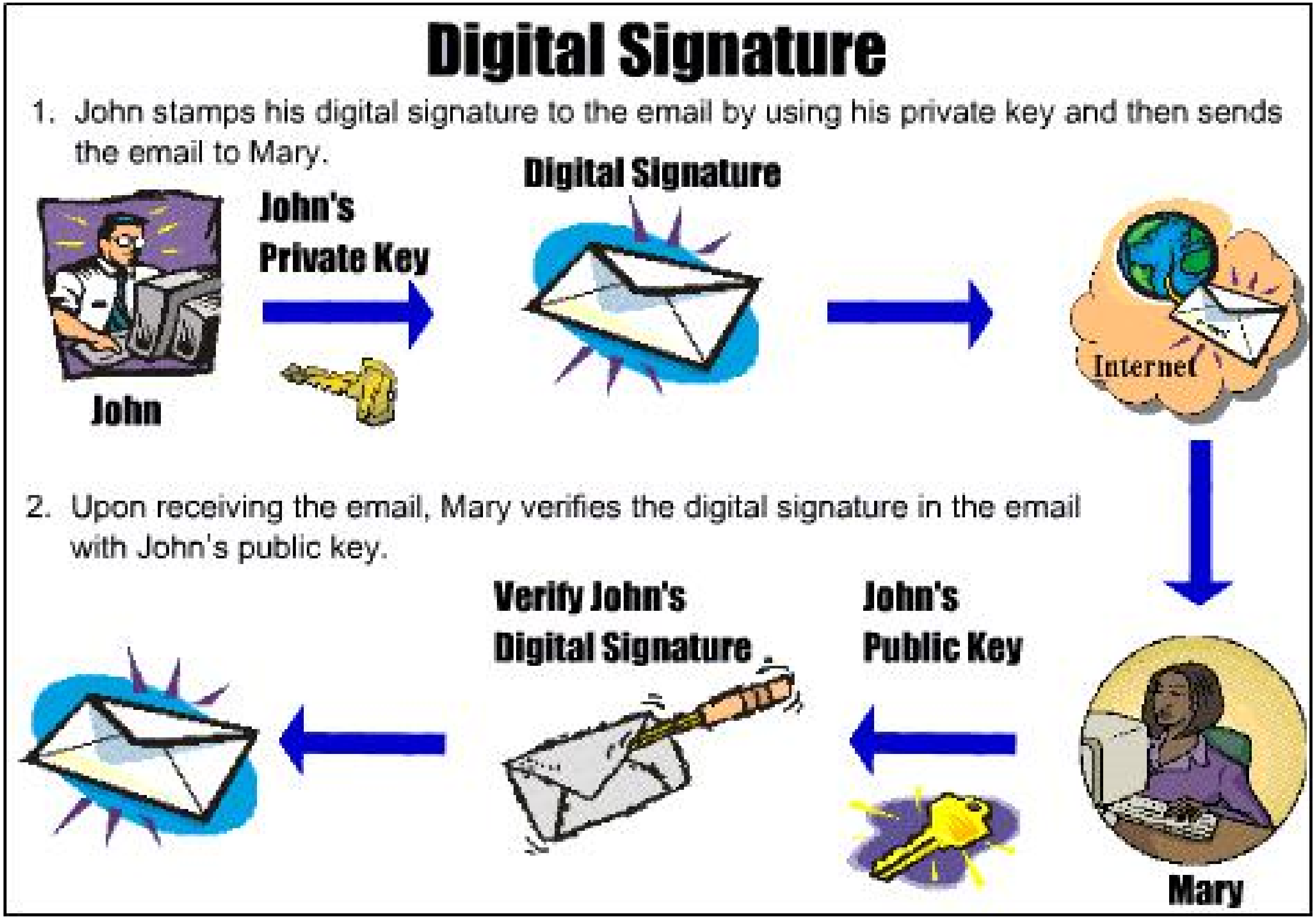
Digital Signatures

A digital signature is an example of using encryption techniques to provide authentication services in e-commerce applications.

- A **digital signature** is a means of associating a mark unique to an individual with a body of text.
  - Other persons should be able to check that the signature does come from the originator.
- A **digital signature** consists of a string of symbols.
  - Must be different for each use: achieved by making each digital signature a function of the message that it is signing, together with a time stamp.
  - To be unique to each signer and counterfeitproof: achieved by making each digital signature dependent on some secret number that is unique to the signer.
- Public key techniques are the means creating digital signatures.

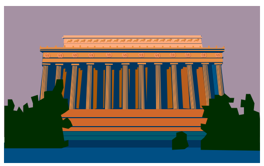


Digital Signatures



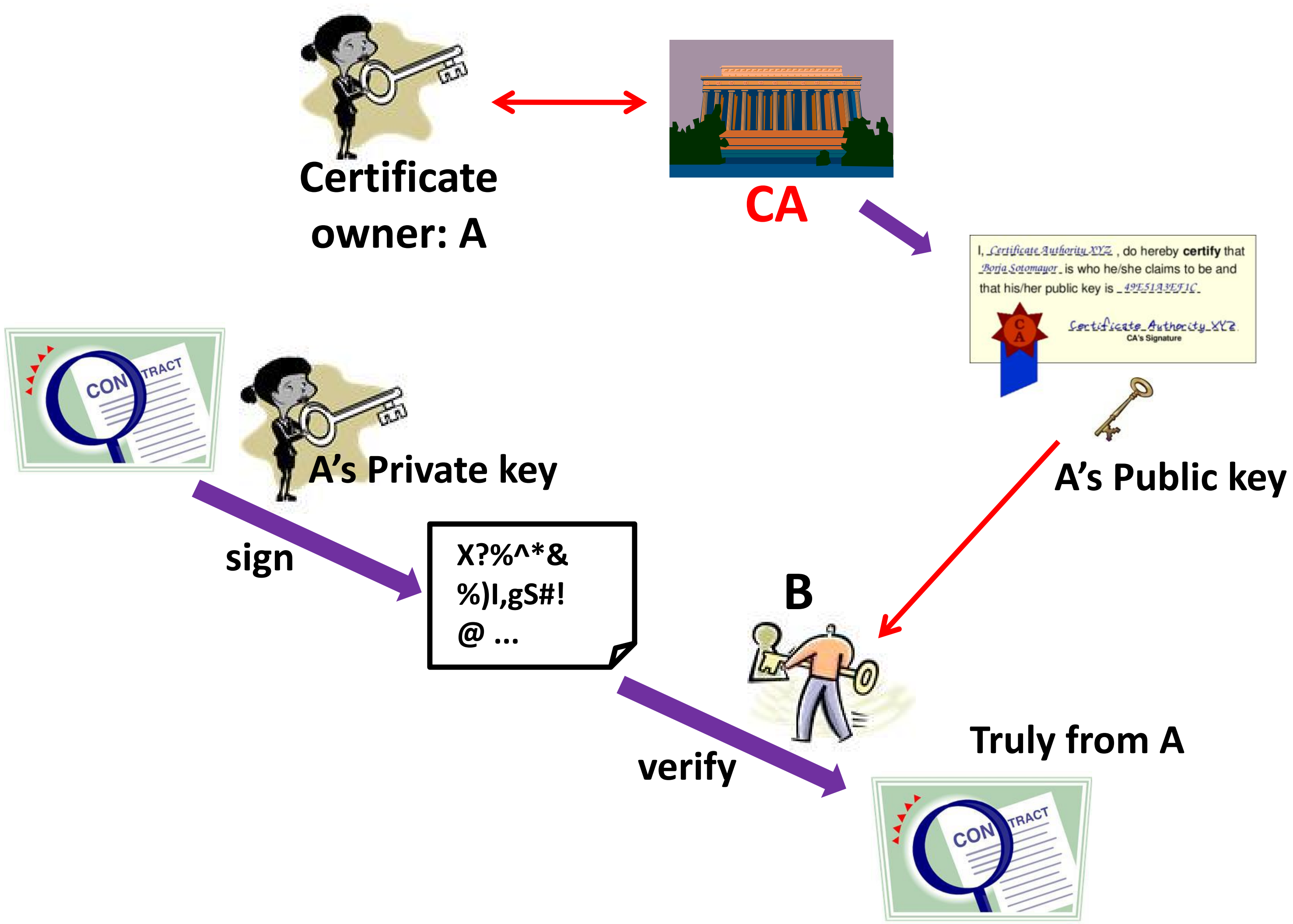
Digital Certificates

- A **digital certificate** is used to combine the value of a public key with the identity of the person or service that holds the corresponding private key into a digitally signed statement.
- Certificates are issued and signed by a certification authority (CA).



certification authority

Digital Certificate



Certificate owner: A
CA
I, Certificate Authority XYZ, do hereby certify that Borja Sotomayor is who he/she claims to be and that his/her public key is 49E51A3EF1C
Certificate Authority XYZ
CA's Signature
A's Private key
A's Public key
CON TRACT
sign
X?%^*&
%)I,gS#!
@ ...
B
verify
Truly from A
CON TRACT

Digital Certificates

- The information included in the certificate
  - The certificate owner information
  - The public key of the owner
  - The date of issue of the certificate
  - The validity period specified by ‘Valid From’ and ‘Valid To’
  - Issuer identifier
  - The digital signature of the issuing CA
- All the information is encoded through a message-digest function, which creates the digital signature
- The signature certifies that the association between the certificate owner and public key is valid.

Challenges of Database Security

- Data Quality
  - Need techniques and solutions to assess the quality of data
- Intellectual Property Rights
  - Watermarking techniques for relational data have been proposed.
- Database Survivability
  - DB systems need to operate and continue their function, even with reduced capabilities, despite disruptive events such as information warfare attacks.
    - ✓Confinement: to prevent damage further spread
    - ✓Damage assessment
    - ✓Reconfiguration: to allow operation to continue in a degraded mode
    - ✓Repair
    - ✓Fault treatment: to identify the weaknesses exploited in the attack