CAN 304
Computer Systems Security

Lecture 12. Database Security

Week 12: 2022-05-13, 14:00-16:00, Friday

Jie Zhang
Department of Communications and Networking
Email: jie.zhang01@xjtlu.edu.cn
Office: EE522

Review of last week

- Intrusion detection
- Firewalls & Intrusion prevention

Learning objectives

- Define and explain SQL injection attacks and countermeasures.
- Explain how inference poses a security threat in database systems.
- Discuss the use of encryption in a database system.

Outline

- Database security
  - Database management system
  - Relational database
  - SQL injection attack
  - Inference
  - Database access control and encryption
- A quick revision

Part 1. Database Security

Databases

- A database is a **structured** set of data held in computer storage and typically accessed or manipulated by means of **specialized software**.

- Not quite exactly in recent years
  - Structured data: e.g., tables, spreadsheets
  - Semi-structured data: e.g., XML and other markup languages
  - Unstructured data: e.g., images, graphics

- Contains the relationships between data items and groups of data items
- Can sometimes contain sensitive data that needs to be secured

DBMS

- Database management system (DBMS) is a suite of programs for:

Constructing and maintaining the database

Offering ad hoc query facilities to multiple users and applications

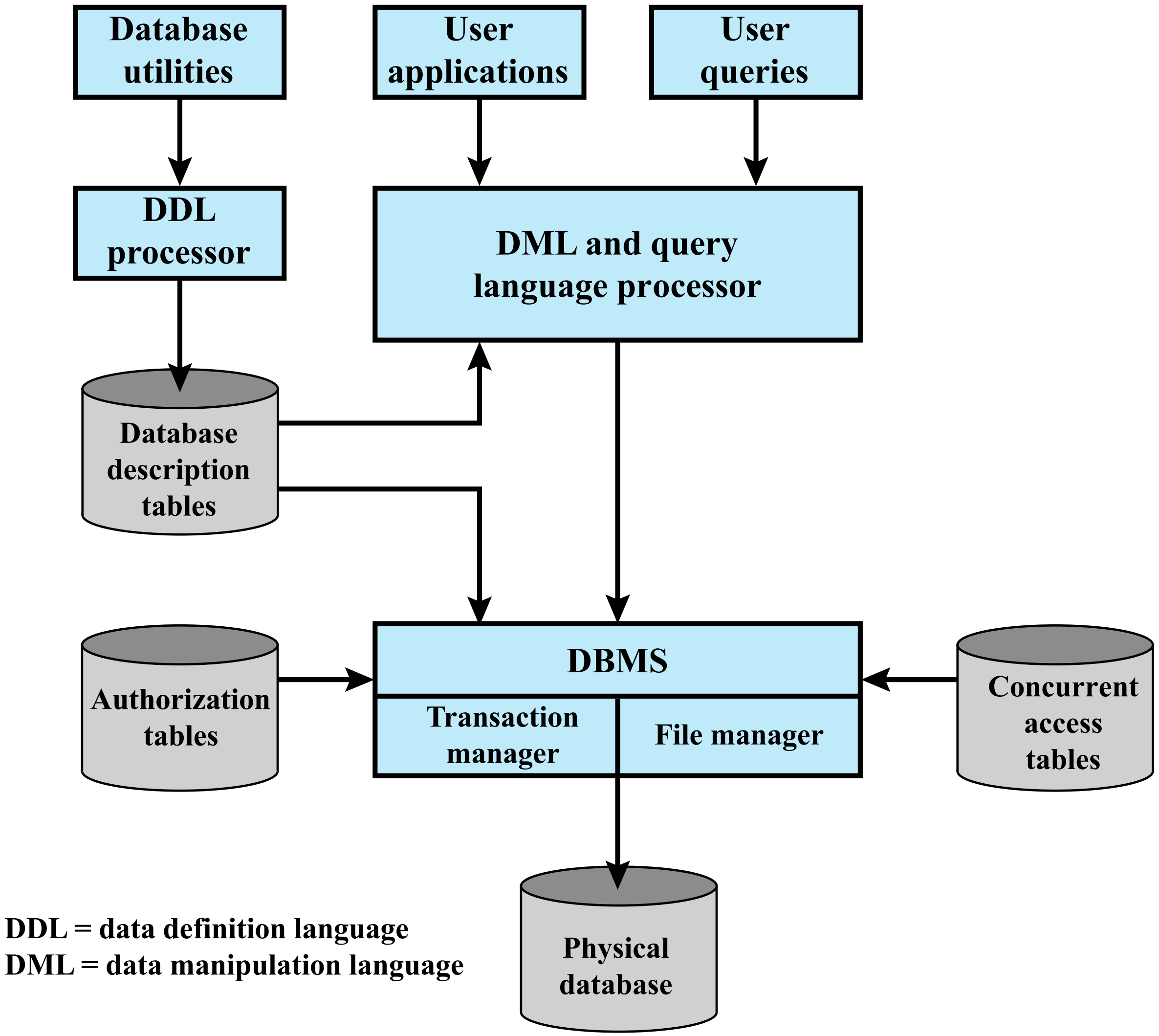
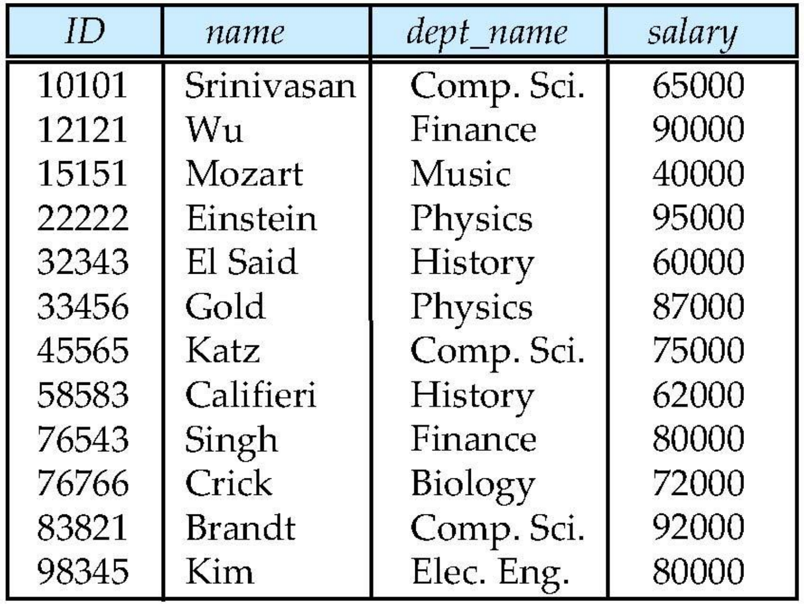


Figure 5.1 DBMS Architecture

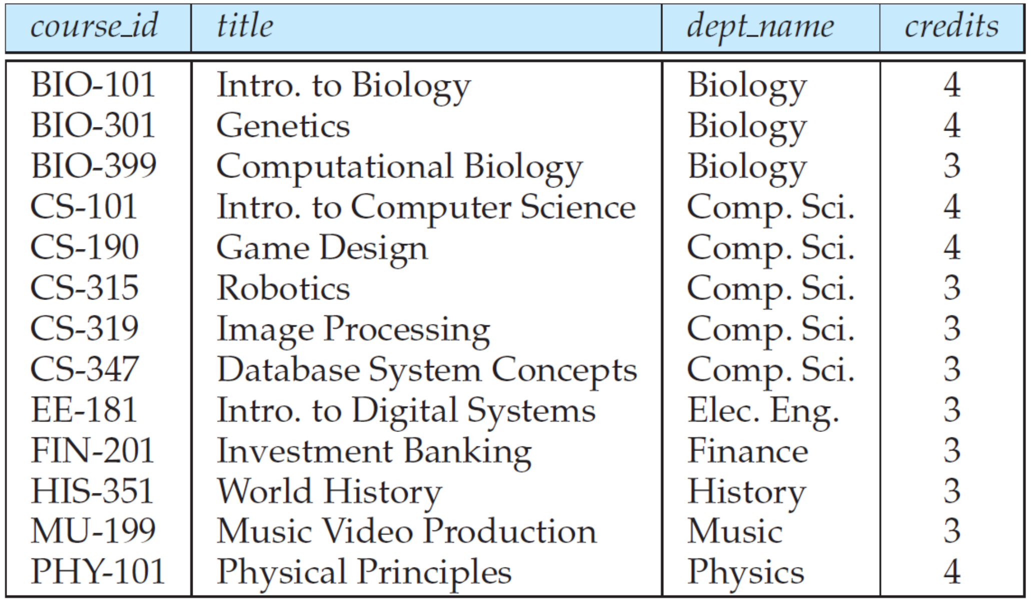
Relational Databases

- A relational database consists of a collection of tables, each of which is assigned a unique name.
- Tables: represent both data and the relationships among those data

- E.g., the instructor table and the course table in a university's database



ID	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000



course_id	*title*	*dept_name*	*credits*
BIO-101	Intro. to Biology	Biology	4
BIO-301	Genetics	Biology	4
BIO-399	Computational Biology	Biology	3
CS-101	Intro. to Computer Science	Comp. Sci.	4
CS-190	Game Design	Comp. Sci.	4
CS-315	Robotics	Comp. Sci.	3
CS-319	Image Processing	Comp. Sci.	3
CS-347	Database System Concepts	Comp. Sci.	3
EE-181	Intro. to Digital Systems	Elec. Eng.	3
FIN-201	Investment Banking	Finance	3
HIS-351	World History	History	3
MU-199	Music Video Production	Music	3
PHY-101	Physical Principles	Physics	4

Relational Database Elements

- Basic Terminology for Relational Databases

Formal Name	Common Name	Also Known As
Relation	Table	File
Tuple	Row	Record
Attribute	Column	Field

Primary key

- Uniquely identifies a row
- Consists of one or more column names

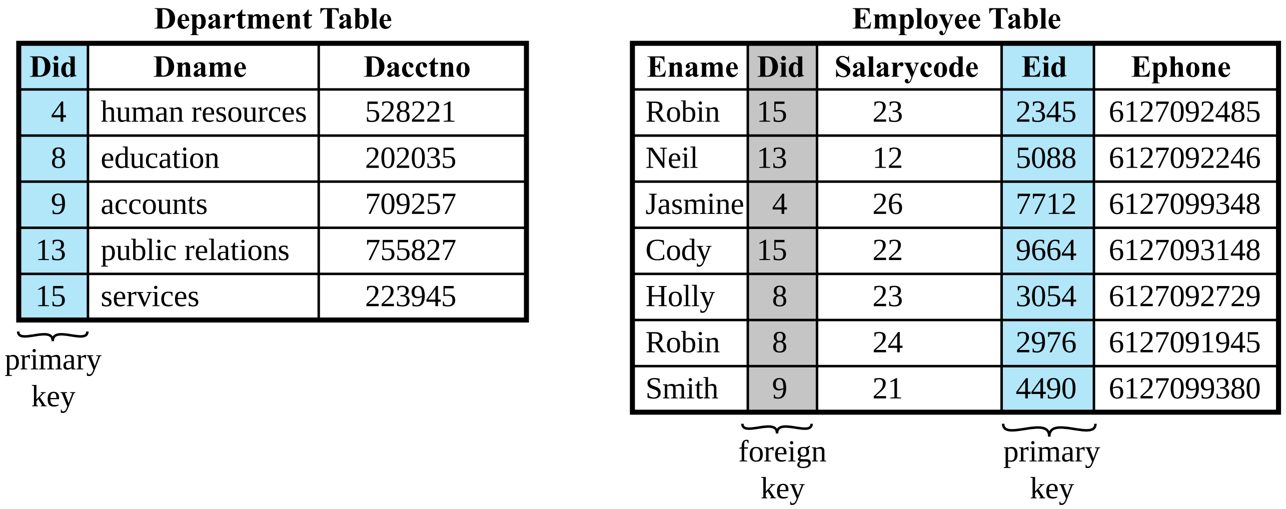
Foreign key

- Links one table to attributes in another

Relational Database: Views

- A view
  - is a “virtual relation” defined by an SQL query
  - conceptually contains the result of the query
  - is not precomputed and stored
  - is computed by executing the query whenever it is used

Relational database example



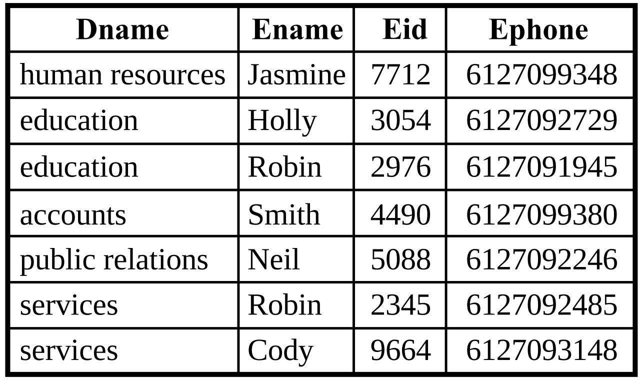
Department Table

Did	Dname	Dacctno
4	human resources	528221
8	education	202035
9	accounts	709257
13	public relations	755827
15	services	223945

Employee Table

Ename	Did	Salarycode	Eid	Ephone
Robin	15	23	2345	6127092485
Neil	13	12	5088	6127092246
Jasmine	4	26	7712	6127099348
Cody	15	22	9664	6127093148
Holly	8	23	3054	6127092729
Robin	8	24	2976	6127091945
Smith	9	21	4490	6127099380

(a) Two tables in a relational database

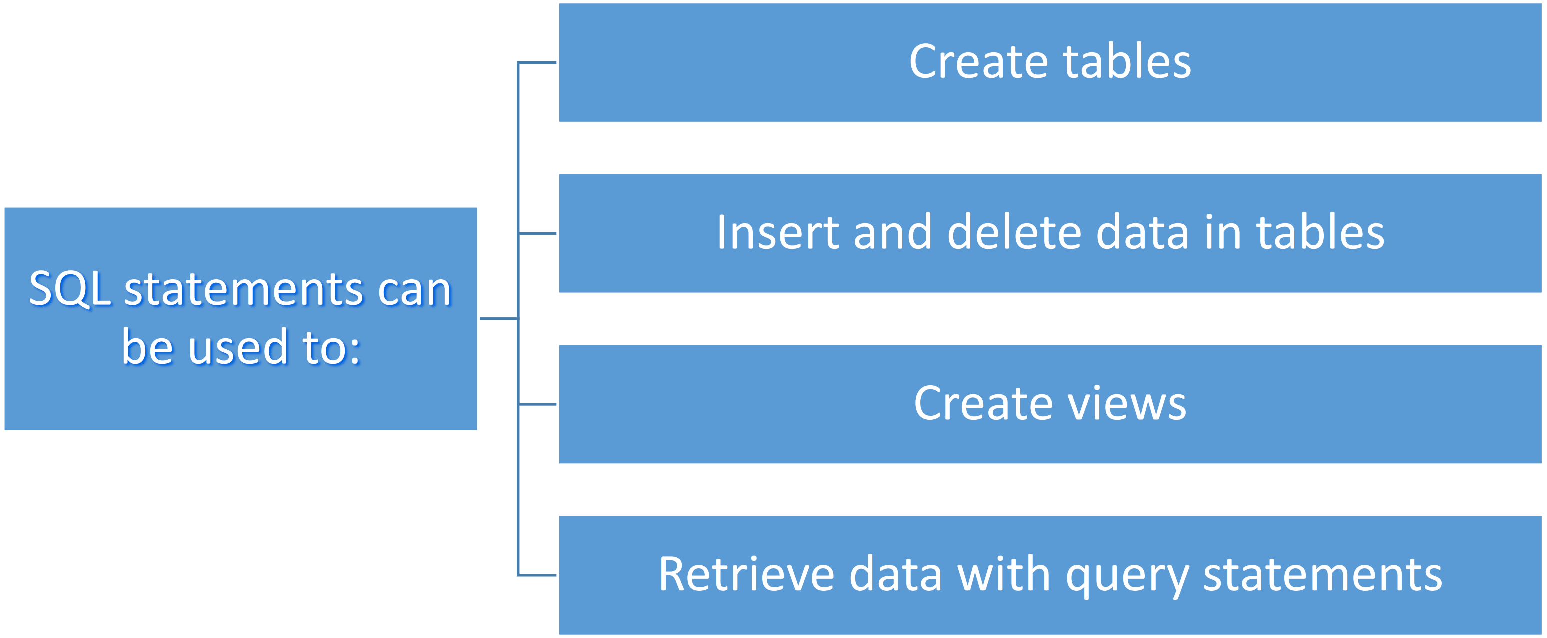


Dname	Ename	Eid	Ephone
human resources	Jasmine	7712	6127099348
education	Holly	3054	6127092729
education	Robin	2976	6127091945
accounts	Smith	4490	6127099380
public relations	Neil	5088	6127092246
services	Robin	2345	6127092485
services	Cody	9664	6127093148

(b) A view derived from the database

Structured Query Language

- Standardized language to define schema, manipulate, and query data in a relational database



SQL statements can be used to:

- Create tables
- Insert and delete data in tables
- Create views
- Retrieve data with query statements

Structured Query Language

- A typical SQL query
  - a_i represents an attribute
  - r_i represents a relation
  - P is a predicate
- The result of an SQL query is a relation

SELECT $a_1, a_2, \dots, a_n$

FROM $r_1, r_2, \dots, r_m$

WHERE P

- Create view command

CREATE VIEW v AS $<$ query expression $>$

Department Table

Did	Dname	Dacctno
4	human resources	528221
8	education	202035
9	accounts	709257
13	public relations	755827
15	services	223945

primary key

Employee Table

Ename	Did	Salarycode	Eid	Ephone
Robin	15	23	2345	6127092485
Neil	13	12	5088	6127092246
Jasmine	4	26	7712	6127099348
Cody	15	22	9664	6127093148
Holly	8	23	3054	6127092729
Robin	8	24	2976	6127091945
Smith	9	21	4490	6127099380

foreign key

primary key

(a) Two tables in a relational database

```
CREATE TABLE department (
   Did INTEGER PRIMARY KEY,
   Dname CHAR (30),
   Dacctno CHAR (6) )
CREATE TABLE employee (
   Ename CHAR (30),
   Did INTEGER,
   SalaryCode INTEGER,
   Eid INTEGER PRIMARY KEY,
   Ephone CHAR (10),
   FOREIGN KEY (Did) REFERENCES department (Did) )
```

```
SELECT Ename, Eid, Ephone
   FROM Employee
   WHERE Did = 15
```

Dname	Ename	Eid	Ephone
human resources	Jasmine	7712	6127099348
education	Holly	3054	6127092729
education	Robin	2976	6127091945
accounts	Smith	4490	6127099380
public relations	Neil	5088	6127092246
services	Robin	2345	6127092485
services	Cody	9664	6127093148

(b) A view derived from the database

```
CREATE VIEW newtable (Dname, Ename, Eid, Ephone)
AS SELECT D.Dname E.Ename, E.Eid, E.Ephone
FROM Department D Employee E
WHERE E.Did = D.Did
```

SQL Injection Attacks (SQLi)

- One of the most prevalent and dangerous network-based security threats
- Designed to exploit the nature of Web application pages
- Sends malicious SQL commands to the database server

- Most common attack goal is bulk extraction of data

- Depending on the environment SQL injection can also be exploited to:
  - Modify or delete data
  - Execute arbitrary operating system commands
  - Launch denial-of-service (DoS) attacks

SQLi Example 1

- Consider a script that build an SQL query by combining predefined strings with text entered by a user:

```
var ShipCity;
ShipCity = Request.form (“ShipCity”);
var sql = “select * from OrdersTable where ShipCity = ‘” + ShipCity + “’”;
```

- The intention of the script’s designer is that a user will enter the name of a city.
- When the script is executed, the user is prompted to enter a city

SQLi Example 1

- Suppose the user input is $User_Input$

```
SELECT *
FROM OrdersTable
WHERE ShipCity = ‘$User_Input$’
```

- $User_Input$ <- Suzhou
- Then, retrieve the rows (all attributes) from the OrdersTable where the orders are being shipped to Suzhou

SQLi Example 1

- Suppose the user input is $User_Input$

```
SELECT *
FROM OrdersTable
WHERE ShipCity = ‘$User_Input$’
```

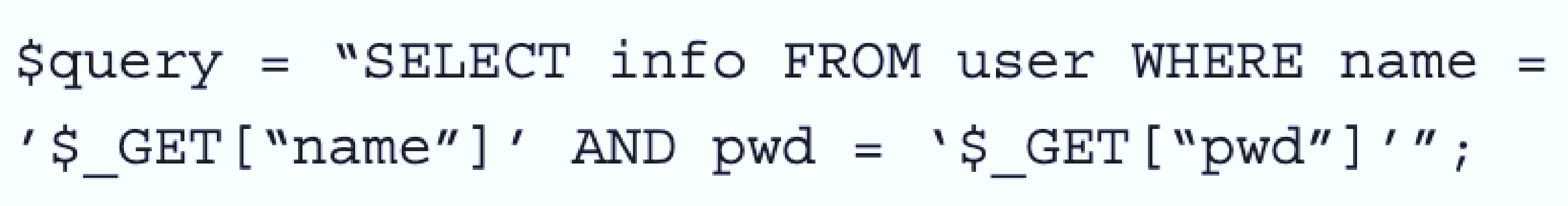
- $User_Input$ <- Suzhou ’; DROP table OrdersTable--

```
SELECT *
FROM OrdersTable
WHERE ShipCity = ‘Suzhou’; DROP table OrdersTable--’
```

- Select all records which are being shipped to Suzhou
- And executes DROP request, which deletes the table

SQLi Example 2

- Consider the script which intends to require the user to enter a valid name and password



```
$query = “SELECT info FROM user WHERE name =
‘$_GET[“name”]’ AND pwd = ‘$_GET[“pwd”]’”;
```

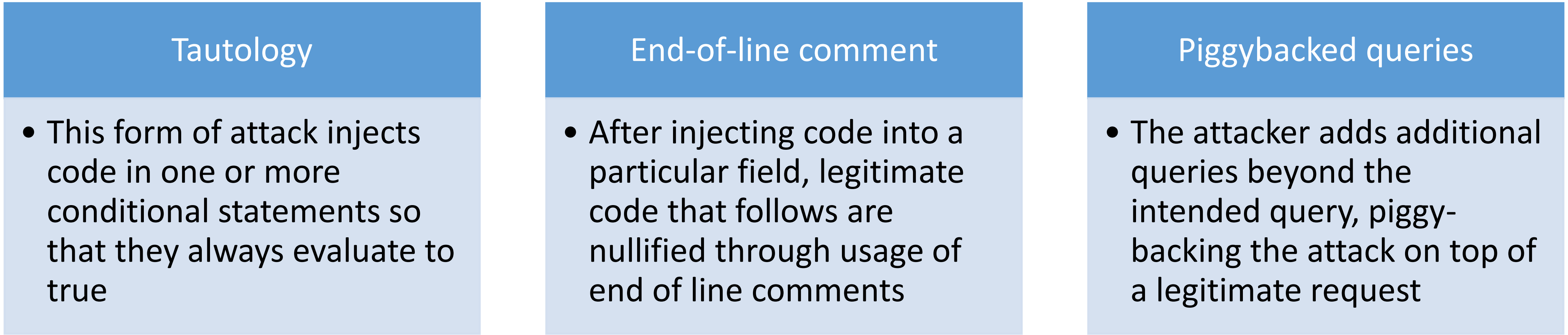
- Suppose the attacker submits “ ’ OR 1=1 --” for the name field.
- The resulting query would look like this:

  SELECT info FROM users WHERE name = ‘ ’ OR 1=1 --’ AND pwd = ‘ ’

Inband Attacks

- Uses the same communication channel for injecting SQL code and retrieving results
- The retrieved data are presented directly in application Web page

- Include:



Tautology

- This form of attack injects code in one or more conditional statements so that they always evaluate to true

End-of-line comment

- After injecting code into a particular field, legitimate code that follows are nullified through usage of end of line comments

Piggybacked queries

- The attacker adds additional queries beyond the intended query, piggy-backing the attack on top of a legitimate request

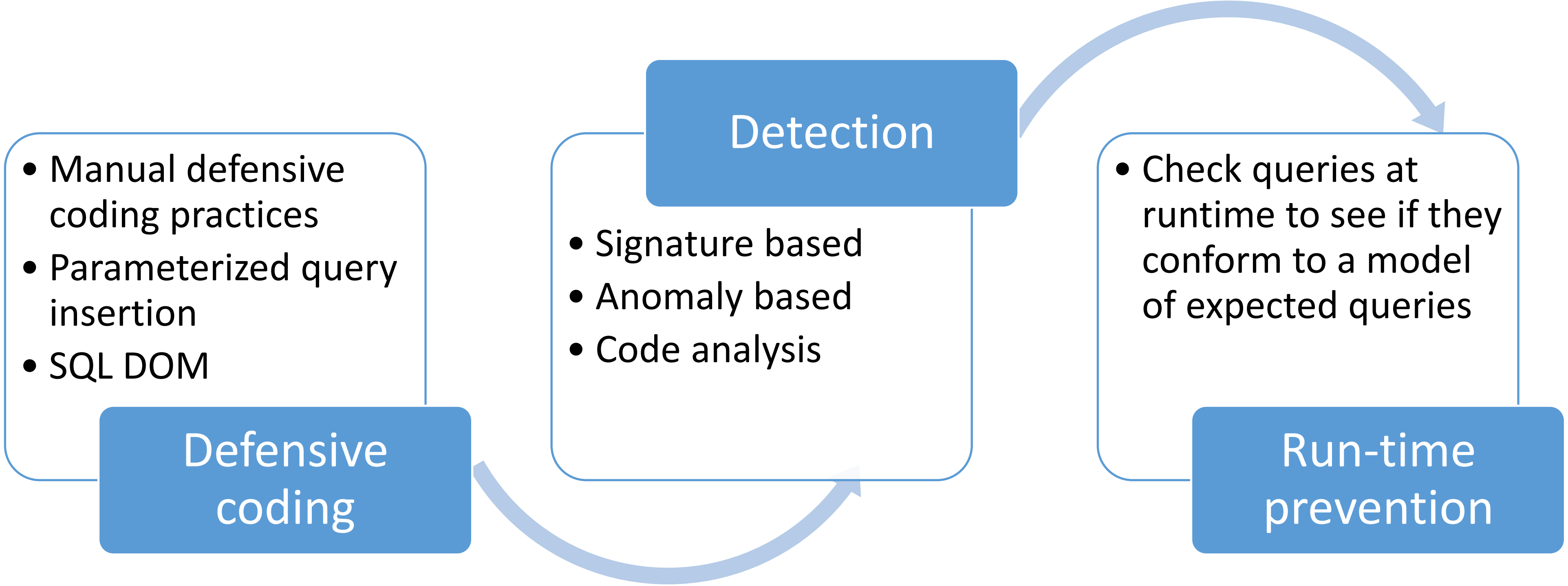
Inferential and Out-of-Band Attacks

- Inferential Attack
  - There is no actual transfer of data, but the attacker is able to reconstruct the information by sending particular requests and observing the resulting behavior of the Website/database server

- Out-of-Band Attack
  - Data are retrieved using a different channel
  - This can be used when there are limitations on information retrieval, but outbound connectivity from the database server is lax

SQLi Countermeasures

- Three types:



Defensive coding

- Manual defensive coding practices
- Parameterized query insertion
- SQL DOM

Detection

- Signature based
- Anomaly based
- Code analysis

Run-time prevention

- Check queries at runtime to see if they conform to a model of expected queries

Database Access Control

- A DBMS can support a range of administrative policies

Centralized administration	Ownership-based administration	Decentralized administration
• Small number of privileged users may grant and revoke access rights	• The creator of a table may grant and revoke access rights to the table	• The owner of the table may grant and revoke authorization rights to other users, allowing them to grant and revoke access rights to the table

SQL Access Controls

- Two commands for managing access rights:
  - Grant
  - Revoke

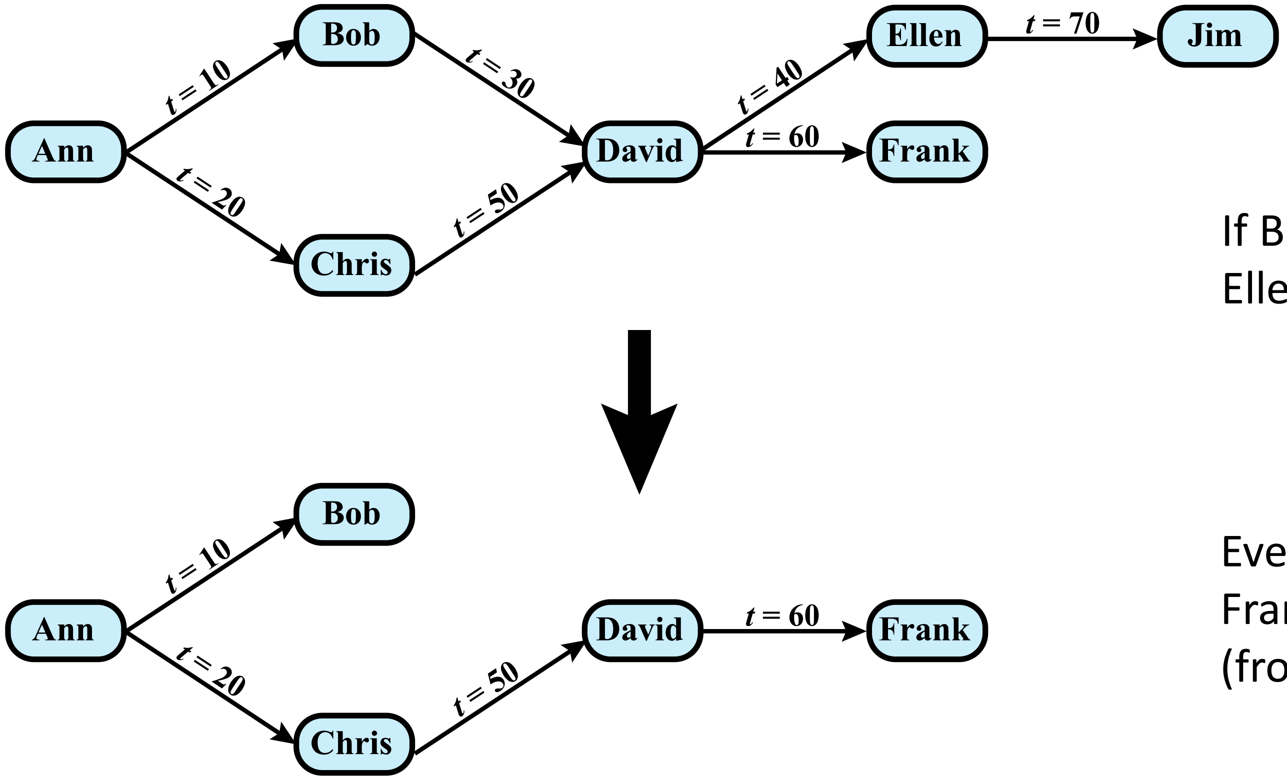
```
GRANT <privilege list>
ON  <relation name or view name>
TO <user/role list>;

REVOKE <privilege list>
ON  <relation name or view name>
FROM <user/role list>;
```

- Typical access rights are:
  - Select
  - Insert
  - Update
  - Delete
  - References

Cascading Authorizations

- Bob Revokes Privilege from David



If Bob have not granted David the privilege, Ellen would have not had the privilege

Even if Bob did not grant David the privilege, Frank would have still granted the privilege (from Chris to David to Frank)

Role-Based Access Control

- Role-based access control eases administrative burden and improves security
- Categories of database users:

Application owner

- An end user who owns database objects as part of an application

End user

- An end user who operates on database objects via a particular application but does not own any of the database objects

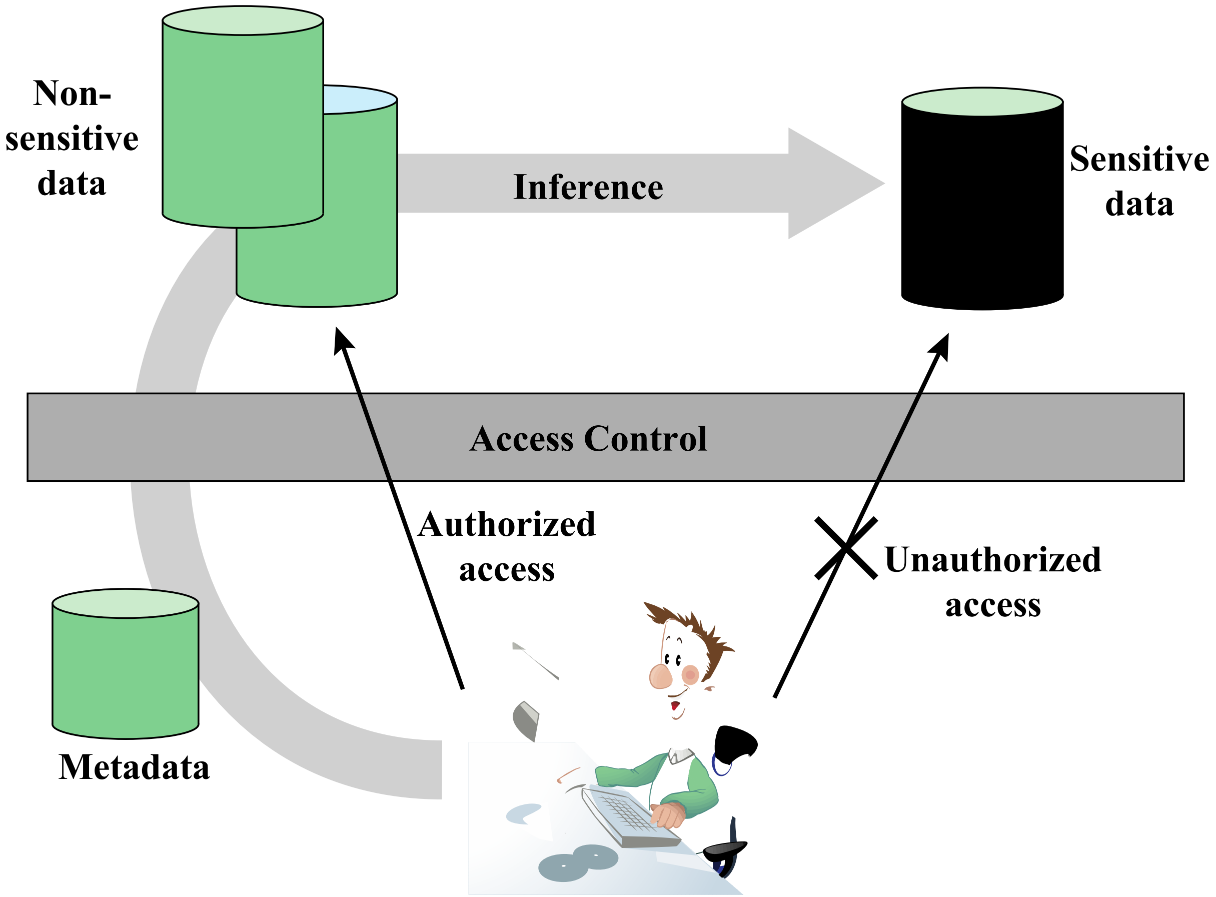
Administrator

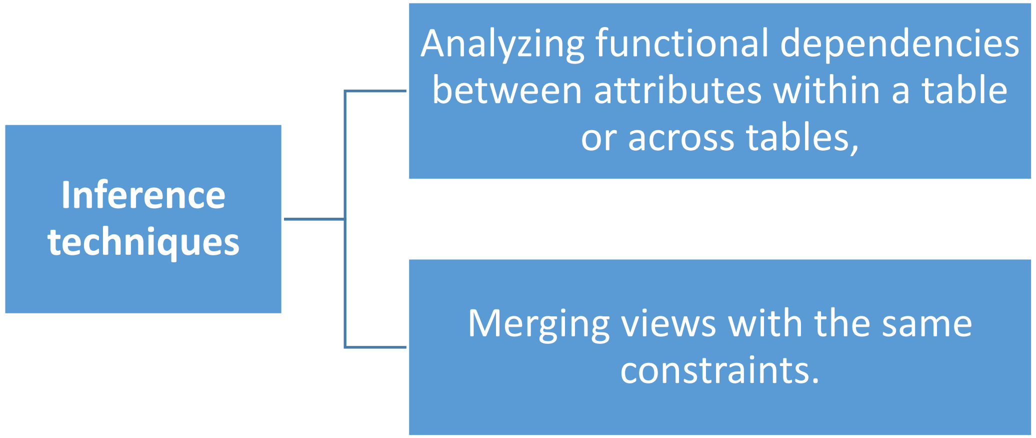
- User who has administrative responsibility for part or all of the database

- A database RBAC facility needs to provide the following capabilities:
  - Create and delete roles
  - Define permissions for a role
  - Assign and cancel assignment of users to roles

Inference

- Indirect Information Access via Inference Channel





Inference techniques

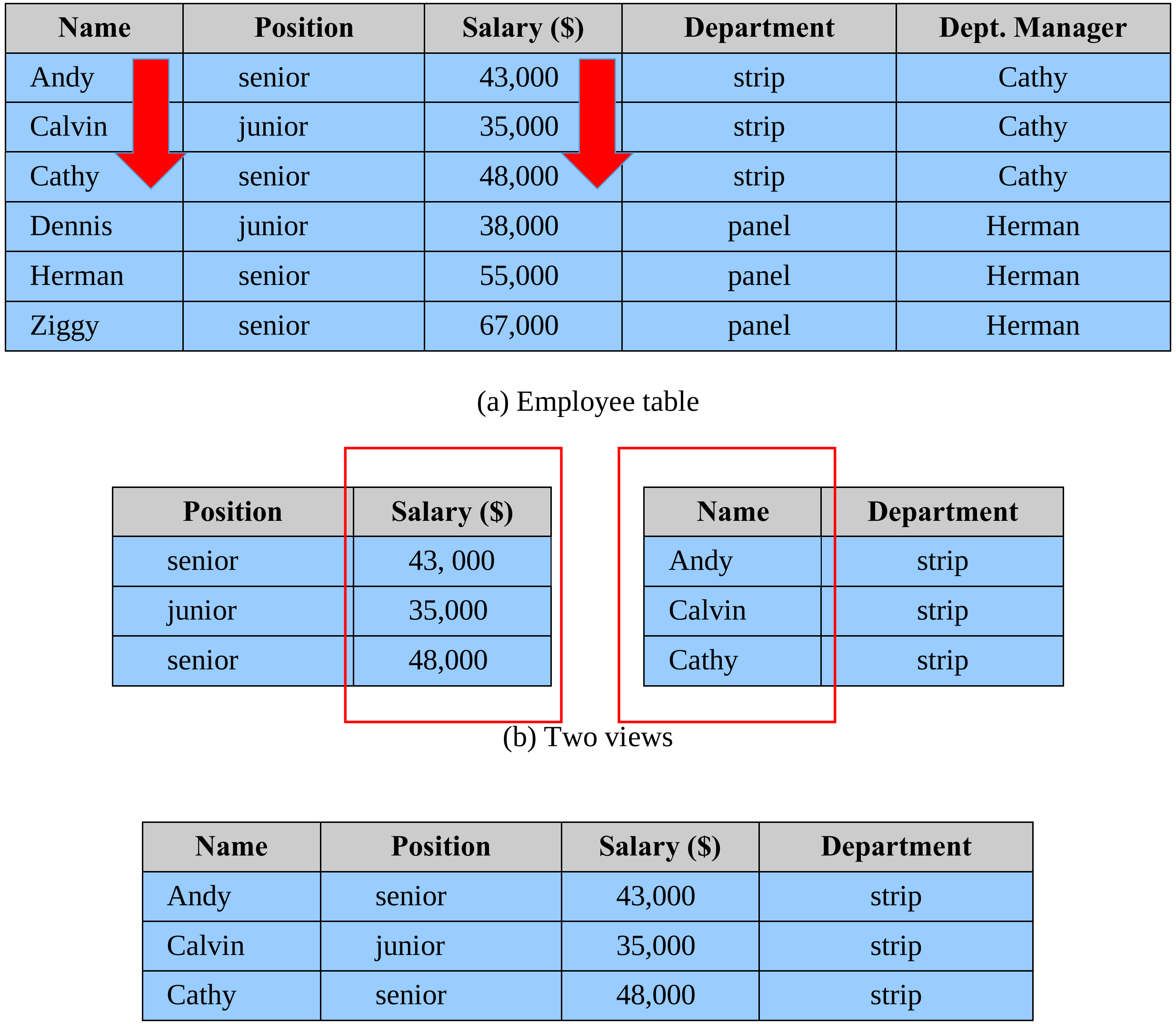
Analyzing functional dependencies between attributes within a table or across tables,

Merging views with the same constraints.

Inference Example

CREATE view V1 AS
SELECT Position, Salary FROM Employee
WHERE Department = ”strip”

CREATE view V2 AS
SELECT Name, Department FROM Employee
WHERE Department = ”strip”



Name	Position	Salary ($)	Department	Dept. Manager
Andy	senior	43,000	strip	Cathy
Calvin	junior	35,000	strip	Cathy
Cathy	senior	48,000	strip	Cathy
Dennis	junior	38,000	panel	Herman
Herman	senior	55,000	panel	Herman
Ziggy	senior	67,000	panel	Herman

(a) Employee table

Position	Salary ($)
senior	43, 000
junior	35,000
senior	48,000

Name	Department
Andy	strip
Calvin	strip
Cathy	strip

(b) Two views

Name	Position	Salary ($)	Department
Andy	senior	43,000	strip
Calvin	junior	35,000	strip
Cathy	senior	48,000	strip

Inference Detection

- Inference detection during database design
  - Approach removes an inference channel by altering the database structure or by changing the access control regime to prevent inference
  - Techniques in this category often result in unnecessarily **stricter access controls** that reduce availability

- Inference detection at query time
  - Approach seeks to eliminate an inference channel violation during a query or series of queries
  - If an inference channel is detected, the query is denied or altered

Database Encryption

- Encryption becomes the last line of defense in database security
  - Can be applied to the entire database, at the record level, the attribute level, or level of the individual field

- Two disadvantages to encryption:
  - Key management
    - Authorized users must have access to the decryption key for the data for which they have access
  - Inflexibility
    - When part or all of the database is encrypted it becomes more difficult to perform record searching

A Database Encryption Scheme

- **Data owner** - organization that produces data to be made available for controlled release

- **User** - human entity that presents queries to the system

- **Client** - frontend that transforms user queries into queries on the encrypted data stored on the server

- **Server** - an organization that receives the encrypted data from a data owner and makes them available for distribution to clients

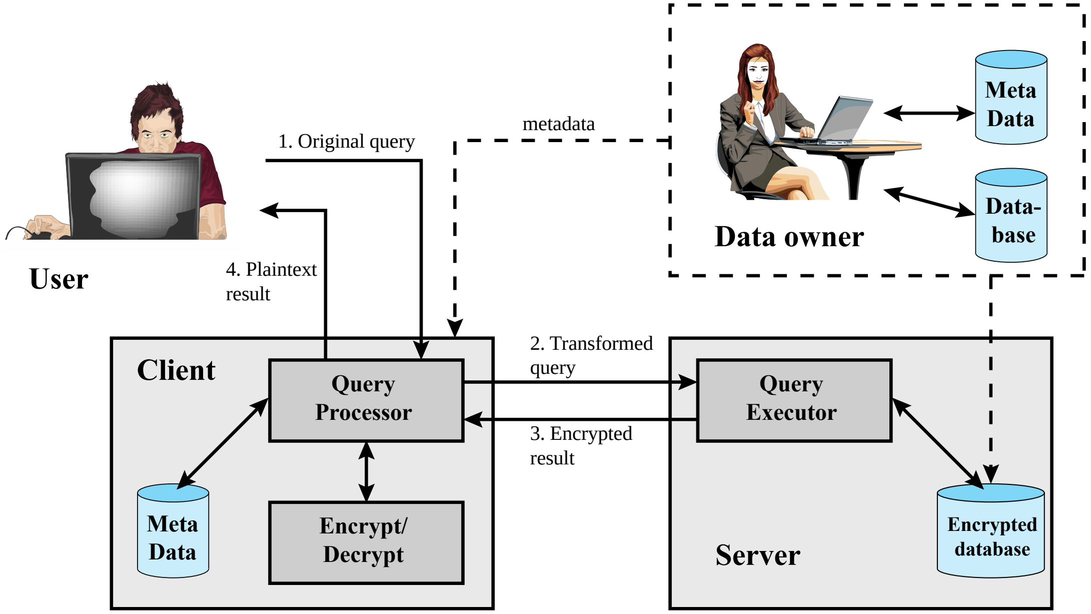


Figure 5.9 A Database Encryption Scheme

A Database Encryption Scheme

- Suppose that each individual item in the database is encrypted separately, all using the same encryption key.

- E.g.

```
SELECT Ename, Eid, Ephone
     FROM Employee
     WHERE Did = 15
```

E(k,15) = 1000110111001110

```
SELECT Ename, Eid, Ephone
     FROM Employee
     WHERE Did = 1000110111001110
```

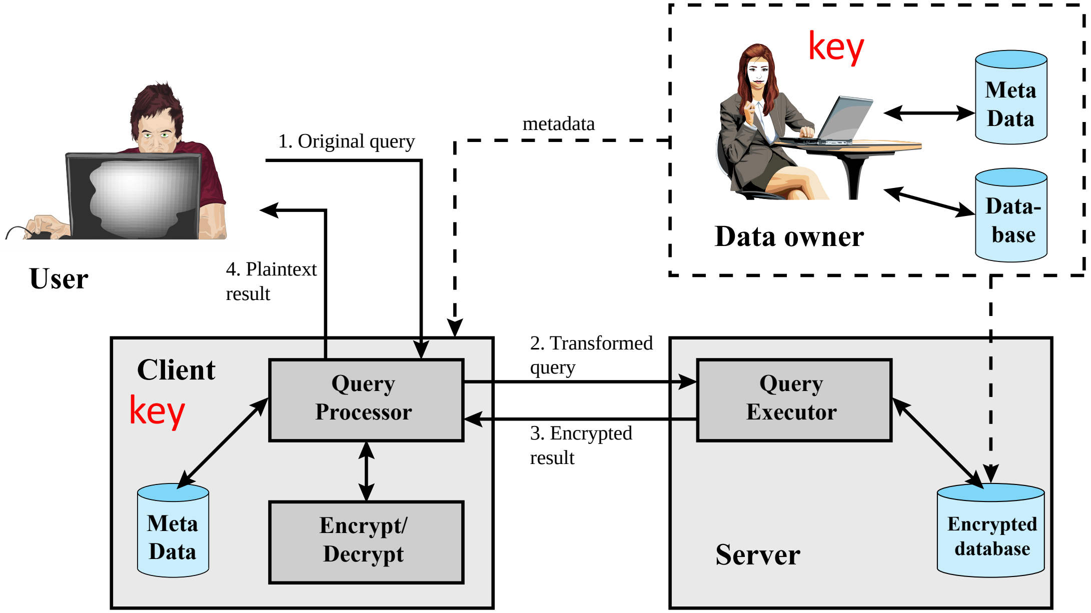


Figure 5.9 A Database Encryption Scheme

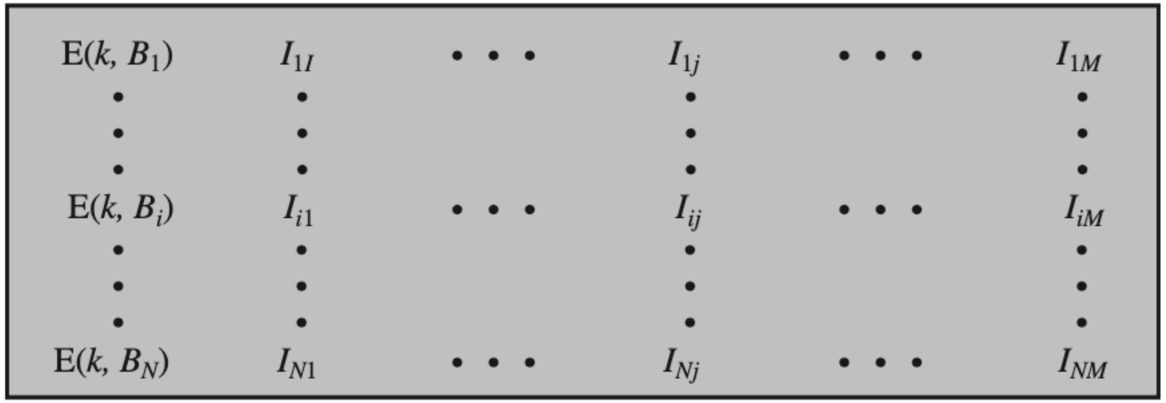
A More Flexible Database Encryption Scheme

- Each record (row) of a table in the database is encrypted as a block.

$$B_i = (x_{i1} \parallel x_{i2} \parallel \cdots \parallel x_{iM})$$
$$E(k, B_i) = E(k, x_{i1} \parallel x_{i2} \parallel \cdots \parallel x_{iM})$$

- To assist in data retrieval, attribute indexes are associated with each table.
- For some or all of the attributes an index value is created.

$$(x_{i1} \parallel x_{i2} \parallel \cdots \parallel x_{iM}) \rightarrow [E(k, B_i), I_{i1}, I_{i2}, \ldots, I_{iM}]$$



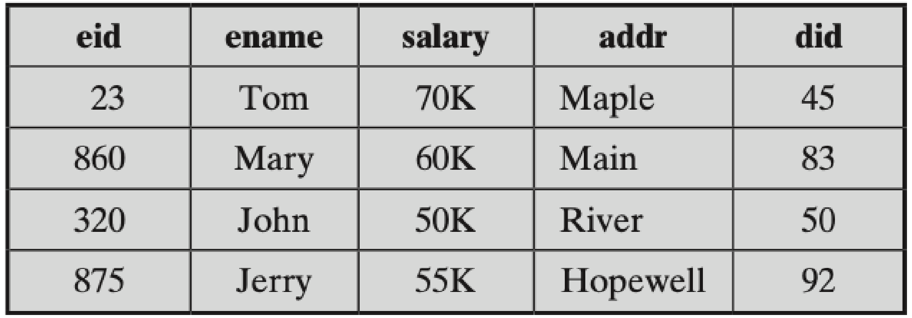
$\mathrm{E}(k, B_1)$	I_{1I}	$\cdots$	I_{1j}	$\cdots$	I_{1M}
$\vdots$	$\vdots$		$\vdots$		$\vdots$
$\mathrm{E}(k, B_i)$	I_{i1}	$\cdots$	I_{ij}	$\cdots$	I_{iM}
$\vdots$	$\vdots$		$\vdots$		$\vdots$
$\mathrm{E}(k, B_N)$	I_{N1}	$\cdots$	I_{Nj}	$\cdots$	I_{NM}

$B_i = (x_{i1} \,||\, x_{i2} \,||\, \ldots \,||\, x_{iM})$

Example

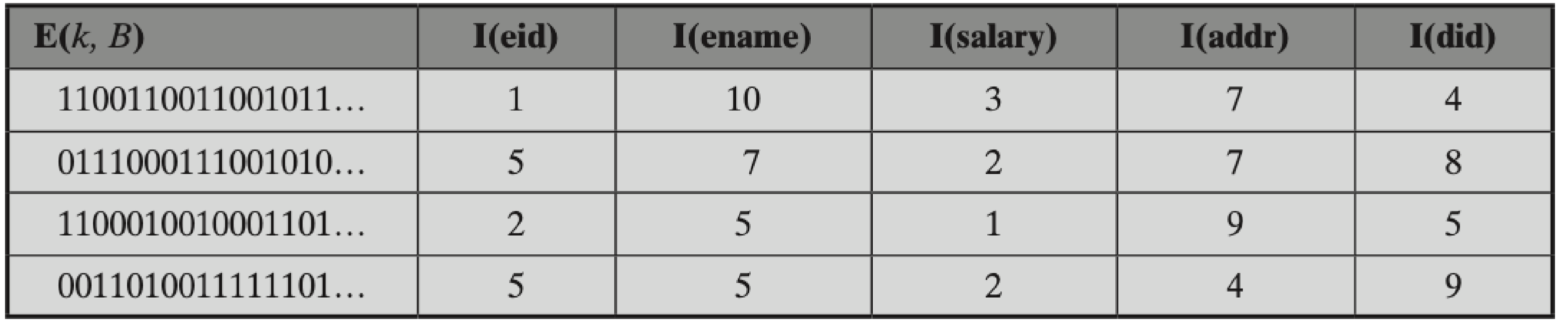
- eid
  - [1, 200]: 1
  - [201, 400]: 2
  - [401, 600]: 3
  - [601, 800]: 4
  - [801, 1000]: 5
- ename
  - A, B: 1
  - C, D: 2
  - …

(a) Employee Table



eid	ename	salary	addr	did
23	Tom	70K	Maple	45
860	Mary	60K	Main	83
320	John	50K	River	50
875	Jerry	55K	Hopewell	92

(b) Encrypted Employee Table with Indexes



E(*k*, *B*)	I(eid)	I(ename)	I(salary)	I(addr)	I(did)
1100110011001011…	1	10	3	7	4
0111000111001010…	5	7	2	7	8
1100010010001101…	2	5	1	9	5
0011010011111101…	5	5	2	4	9

The mapping functions between attribute values and index values constitute metadata that are stored at the client and data owner locations but not at the server.

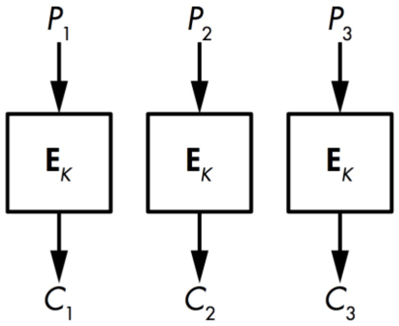
Part 2. Revision

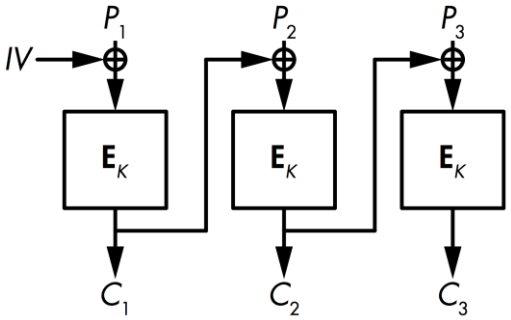
Topics covered so far

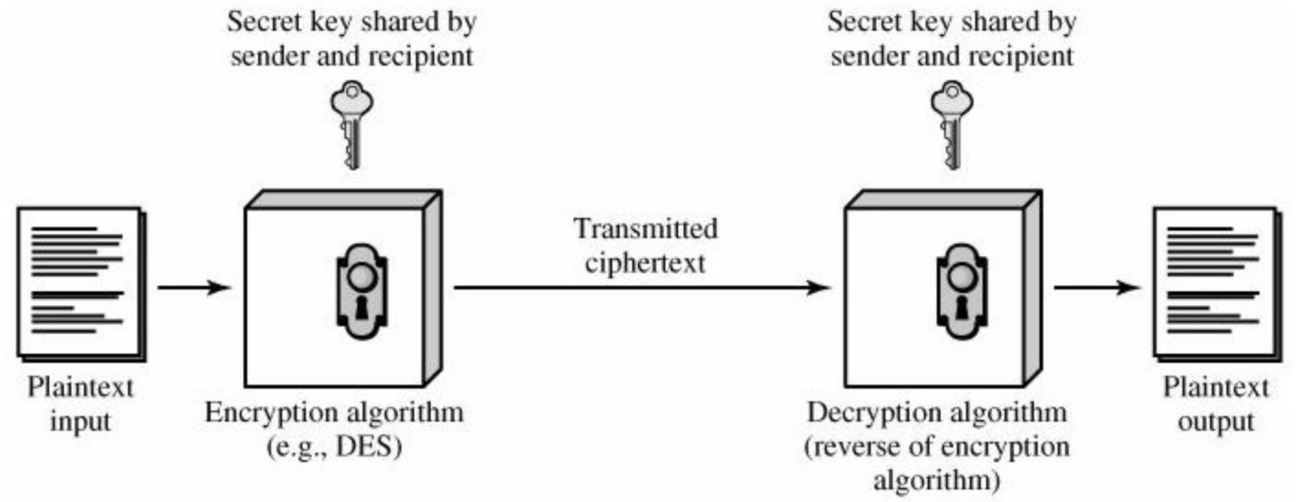
- Foundamentals of cryptography
- Security protocol
- User authentication
- Access control
- Malware
- Dos attacks
- Defense: Intrusion Detection, Firewalls & Intrusion Prevention
- Database security

Fundamentals of cryptography

- Symmetric encryption
  - Understand what is and when to use symmetric ciphers
  - Distinguish stream cipher and block cipher
  - Understand and distinguish ECB and CBC mode
  - Apply symmetric ciphers in security design and implementations.



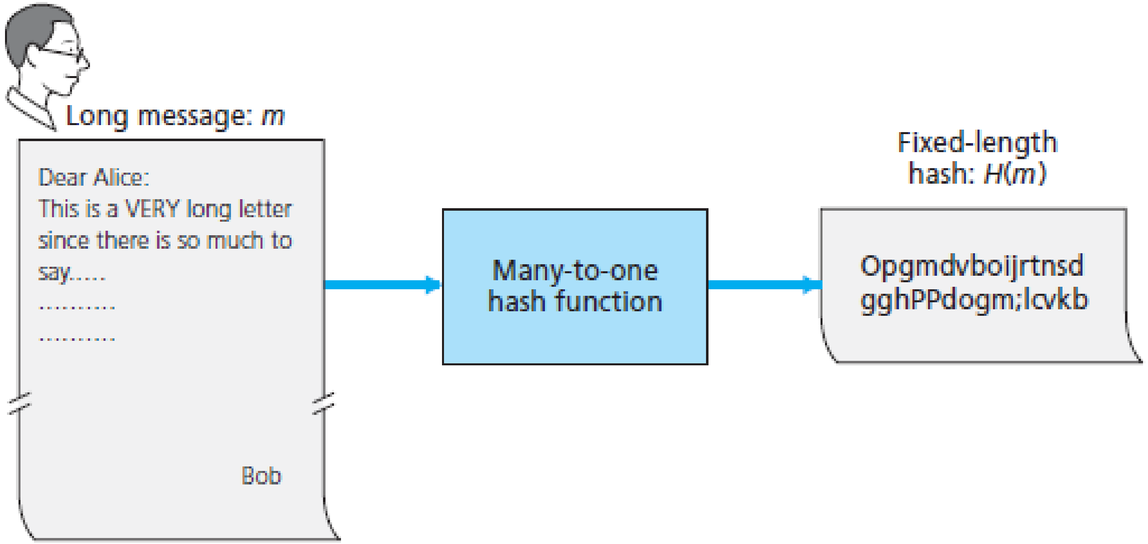




Secret key shared by sender and recipient
Secret key shared by sender and recipient
Transmitted ciphertext
Plaintext input
Encryption algorithm (e.g., DES)
Decryption algorithm (reverse of encryption algorithm)
Plaintext output

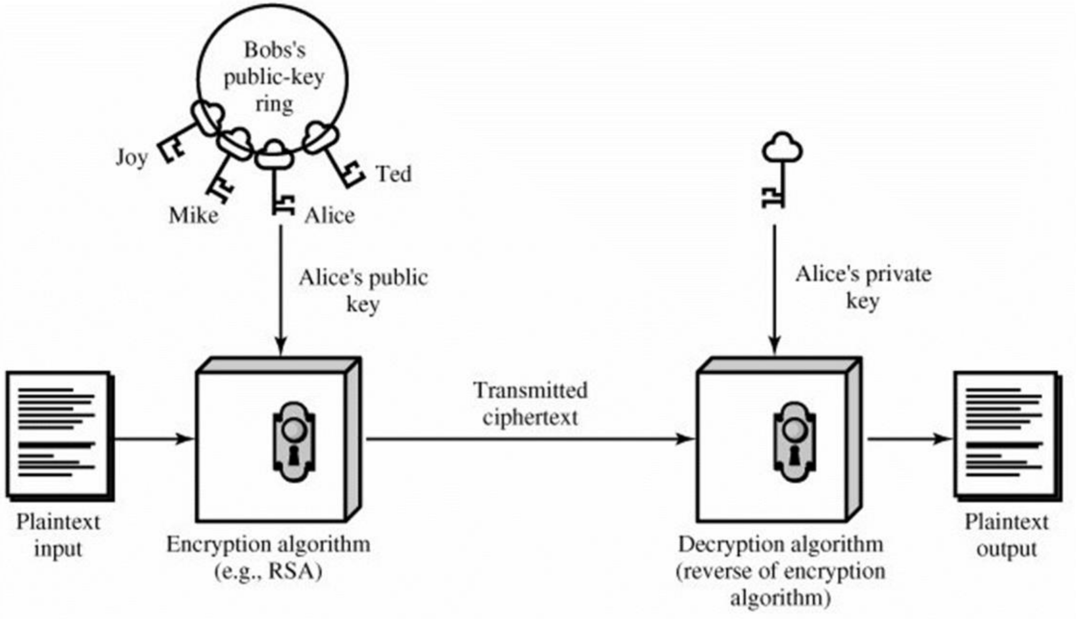
Fundamentals of cryptography

- Message authenticated code and secure hash functions
  - Understand three properties of secure hash functions
  - Apply secure hash functions in security design and implementations.

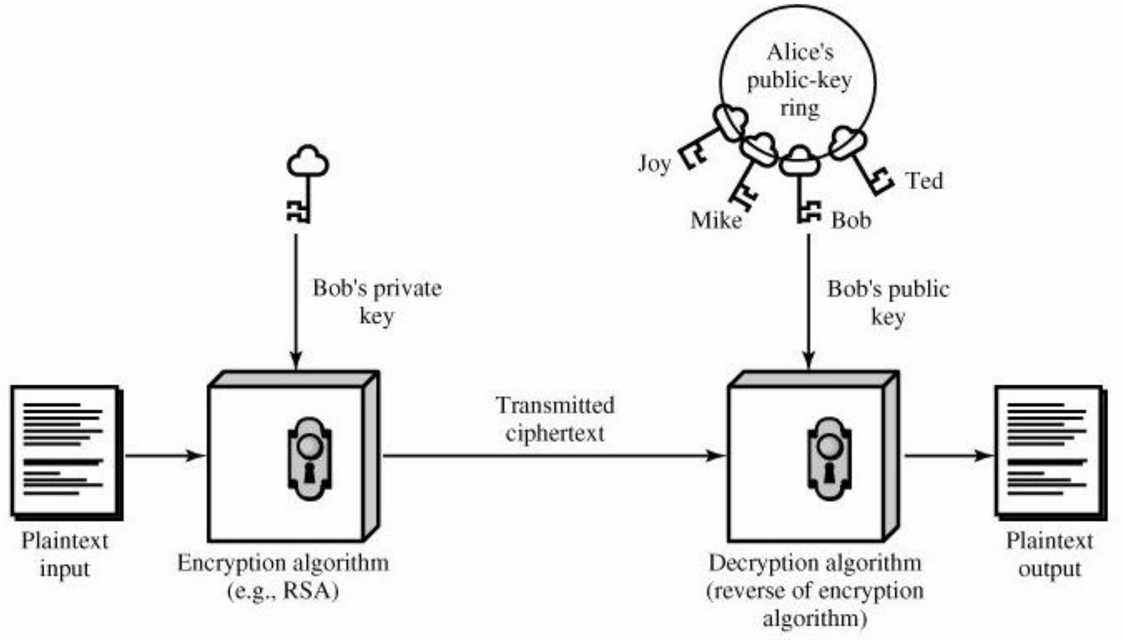

Long message: *m*
Dear Alice:
This is a VERY long letter since there is so much to say.....
....
....
Bob
Many-to-one hash function
Fixed-length hash: *H*(*m*)
Opgmdvboijrtnsd gghPPdogm;lcvkb

Fundamentals of cryptography

- Asymmetric cryptography
  - Understand what is and when to use public-key encryption and digital signature
  - Apply public-key encryption and digital signature in security design and implementations.



(a) Encryption



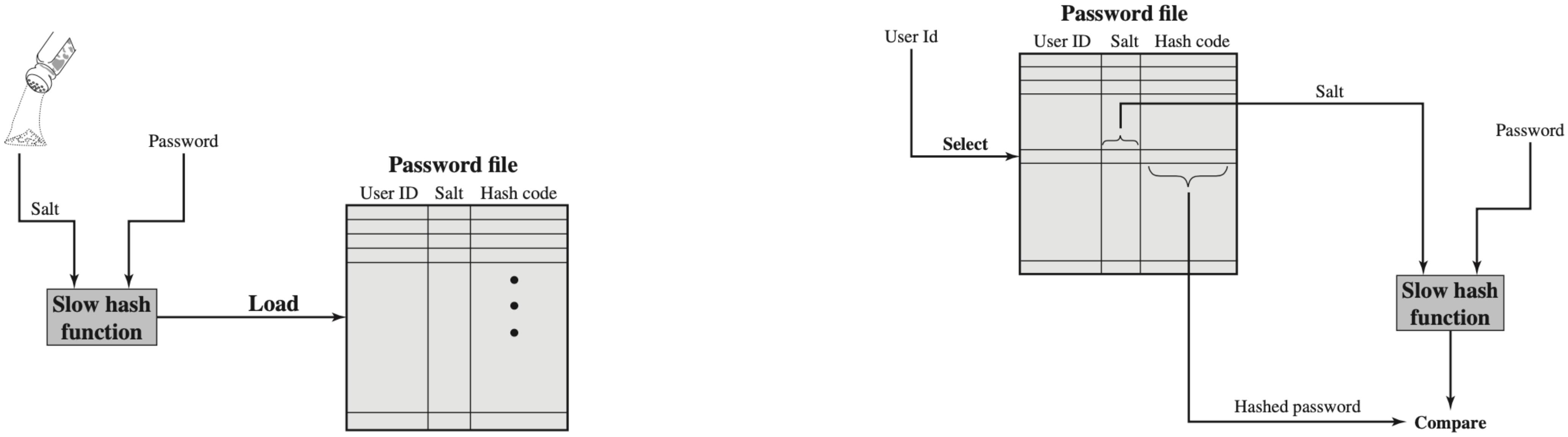
(b) Authentication

Security protocol

- Design a key establishment protocol using cryptographic tools
- Understand potential attacks to a key establishment protocol
- Analyze a given key establishment protocol

User authentication

- Discuss the four general means of user authentication.
- Explain the mechanism by which hashed passwords are used for user authentication.
  - Understand the principle and countermeasures to offline dictionary attacks
- Apply cryptographic tools or no-cryptographic approaches to design a user authentication mechanism.



(a) Loading a new password

(b) Verifying a password

Access control

- Define the three major categories of access control policies.
- Discuss the principal concepts of discretionary access control.
- Discuss the principal concepts of role-based access control.
- Discuss the principal concepts of attribute-based access control.

Malware

- Describe three broad mechanisms malware uses to propagate.
- Learn about the basic operation of viruses, worms, and Trojans.
- Learn about some malware countermeasure elements.

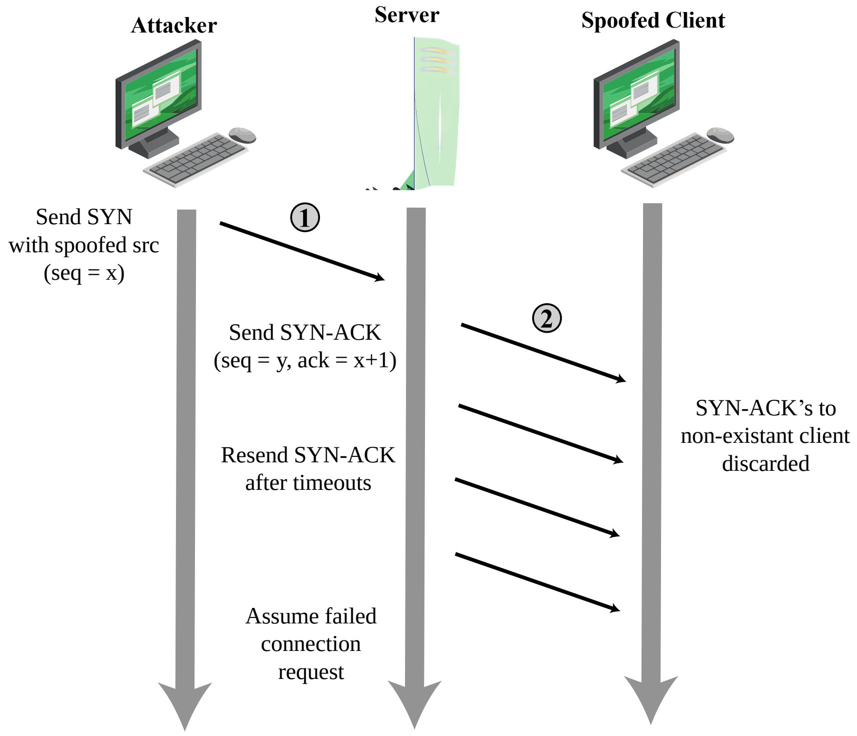
Infection of existing content by **viruses** that is subsequently spread to other systems.

Exploit of software vulnerabilities by **worms** or **drive-by-downloads** to allow the malware to replicate

Social engineering attacks that convince users to bypass security mechanisms to install **Trojans** or to respond to **phishing attacks**.

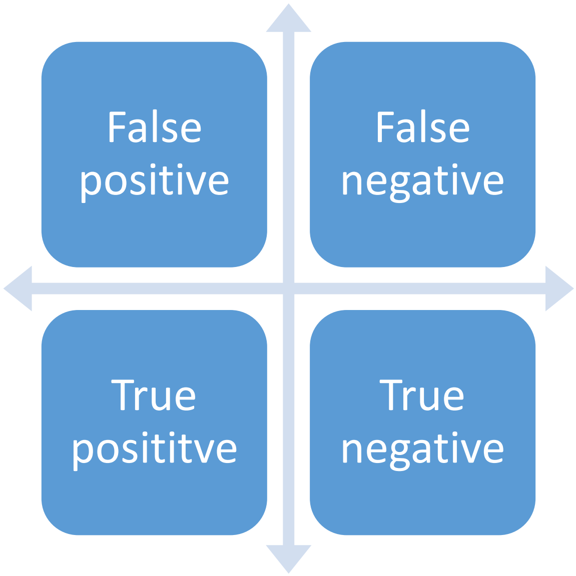
Dos attacks

- Explain the basic concept of a DoS attack.
- Describe the nature of flooding attacks, DDoS attack, reflector and amplifier attacks



Defense: Intrusion Detection, Firewalls & Intrusion Prevention

- Understand anomaly and signature/heuristic approaches for intrusion detection.
- Discuss various types of firewall.

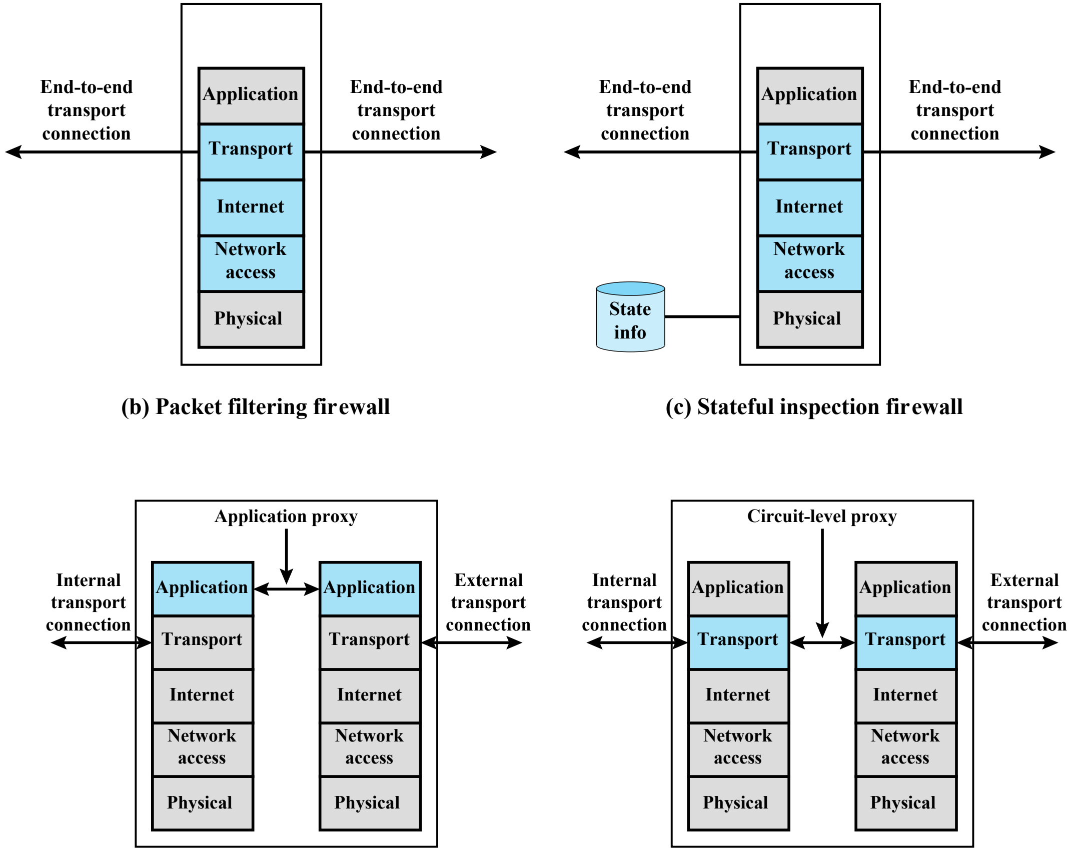


False positive

False negative

True posititve

True negative



(b) Packet filtering firewall

(c) Stateful inspection firewall

(d) Application proxy firewall

(e) Circuit-level proxy firewall

Database security

- Define and explain SQL injection attacks and countermeasures.
- Explain how inference poses a security threat in database systems.
- Discuss the use of encryption in a database system.

Summary

- Database security
  - Database management system
  - Relational database
  - SQL injection attack
  - Inference
  - Database access control and encryption
- A quick revision