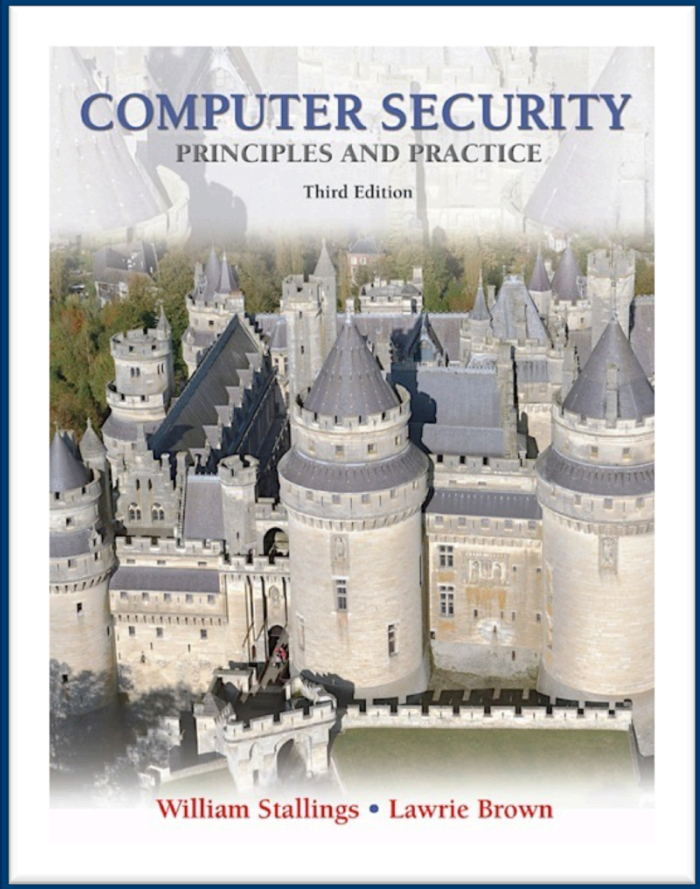
Lecture 15
Database Security

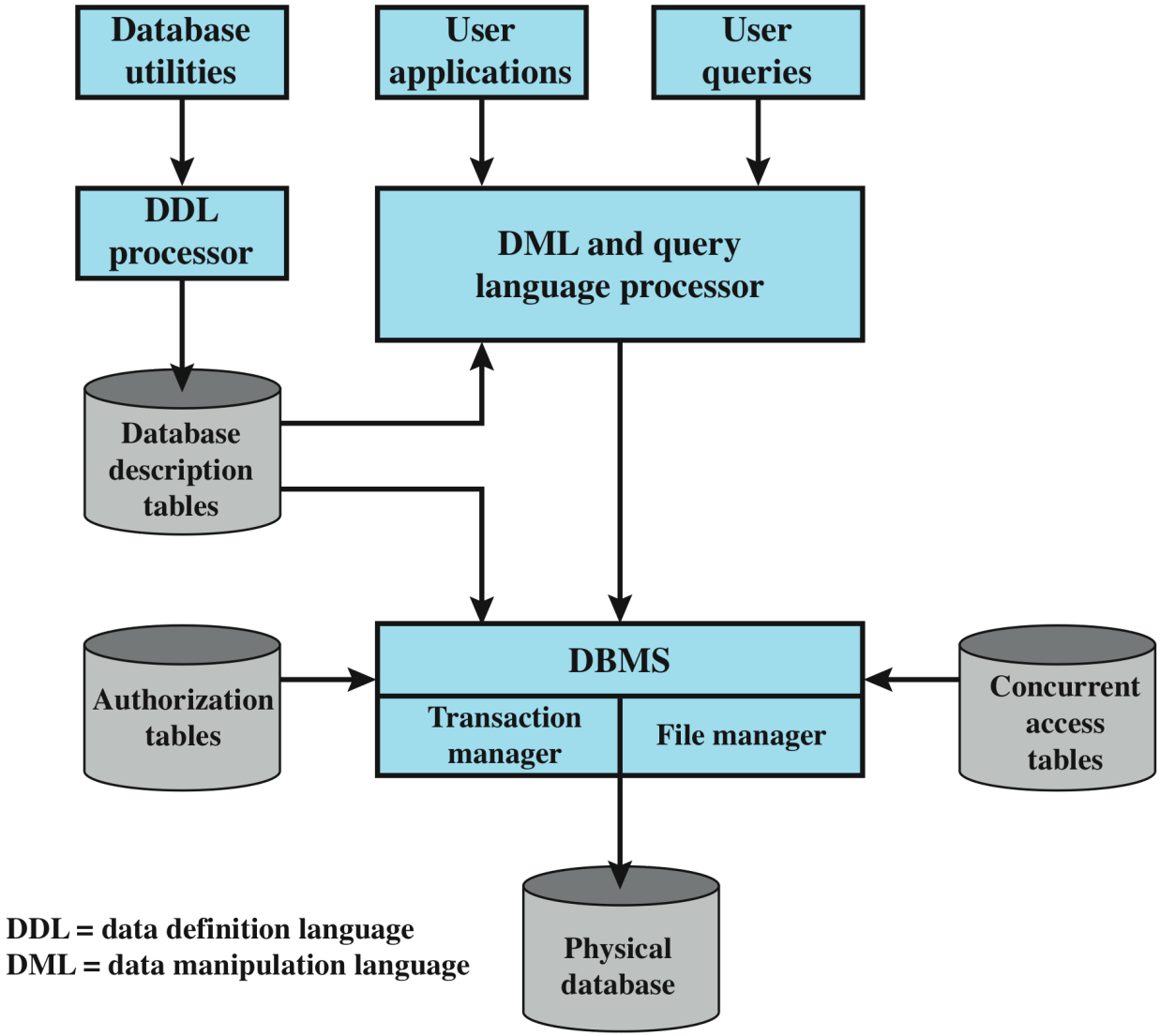


modified from slides of Lawrie Brown

Databases

- structured collection of data stored for use by one or more applications
  - contains the relationships between data items and groups of data items
  - can sometimes contain sensitive data
- database management system (DBMS)
  - suite of programs for constructing and maintaining the database
  - Query language
    - provides a uniform interface to the database

DBMS Architecture



Relational Databases

- table of data consisting of rows and columns
  - each column holds a particular type of data
  - each row contains a specific value for each column
  - ideally has one column where all values are unique, forming an identifier/key for that row
    - enables the creation of multiple tables linked together by a unique identifier that is present in all tables
- use a relational query language to access the database
  - request data that fit a given set of criteria

Relational Database Elements

- 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

view / virtual table

- result of a query that returns selected rows and columns from one or more tables

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

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

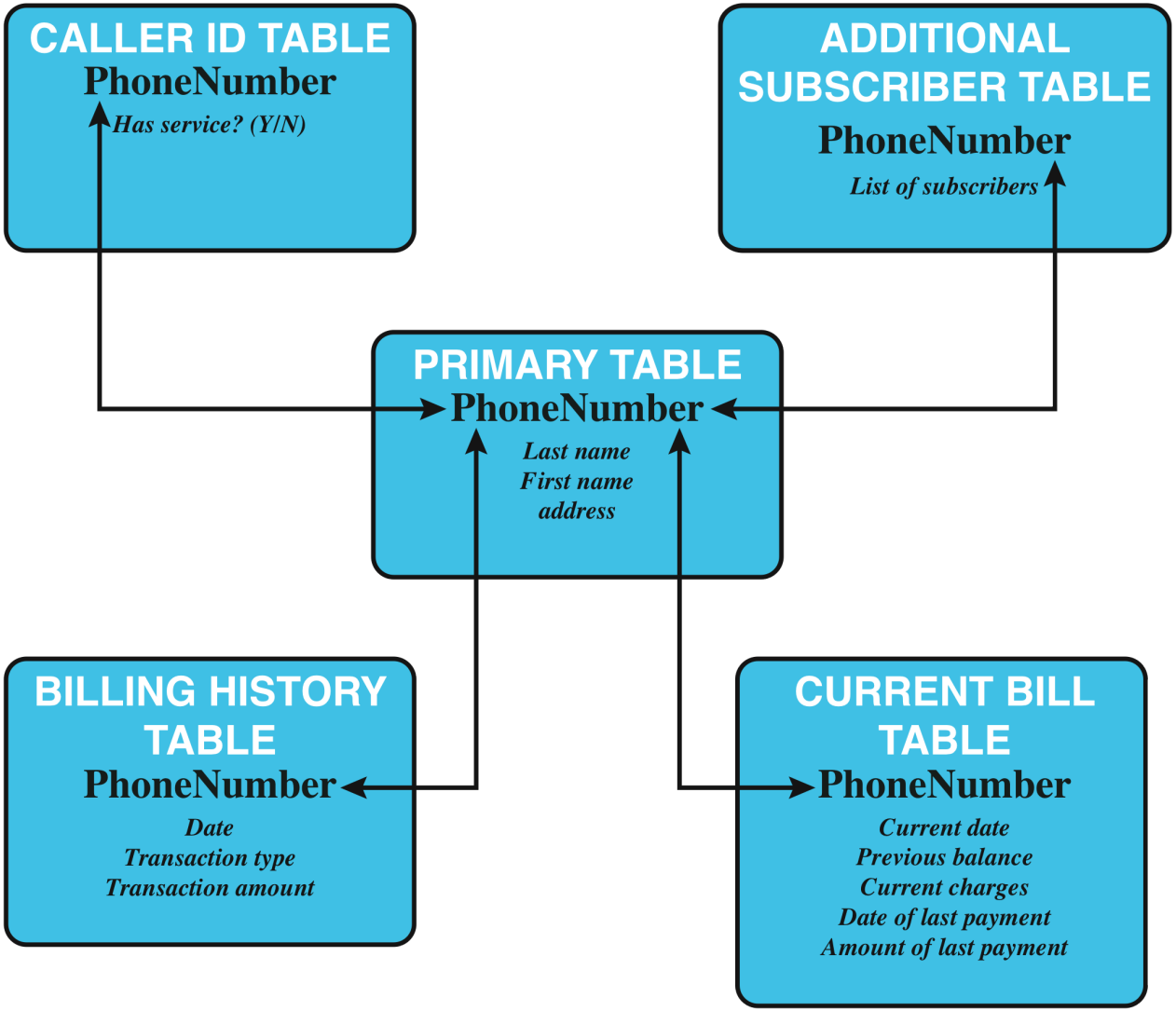
foreign key — primary key

(a) Two tables in a relational database

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

Relational Database Example



Structured Query Language (SQL)

- originally developed by IBM in the mid-1970s
- standardized language to define, manipulate, and query data in a relational database
- several similar versions of ANSI/ISO standard

SQL statements can be used to:

- create tables
- insert and delete data in tables
- create views
- retrieve data with query statements

Structured Query Language (SQL)

```
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) )
```

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

Structured Query Language (SQL)

- The basic command for retrieving information is the SELECT statement. Consider this example:

  – SELECT Ename, Eid, Ephone
    FROM Employee
    WHERE Did 15

- This query returns the Ename, Eid, and Ephone fields from the Employee table for all employees assigned to department 15

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

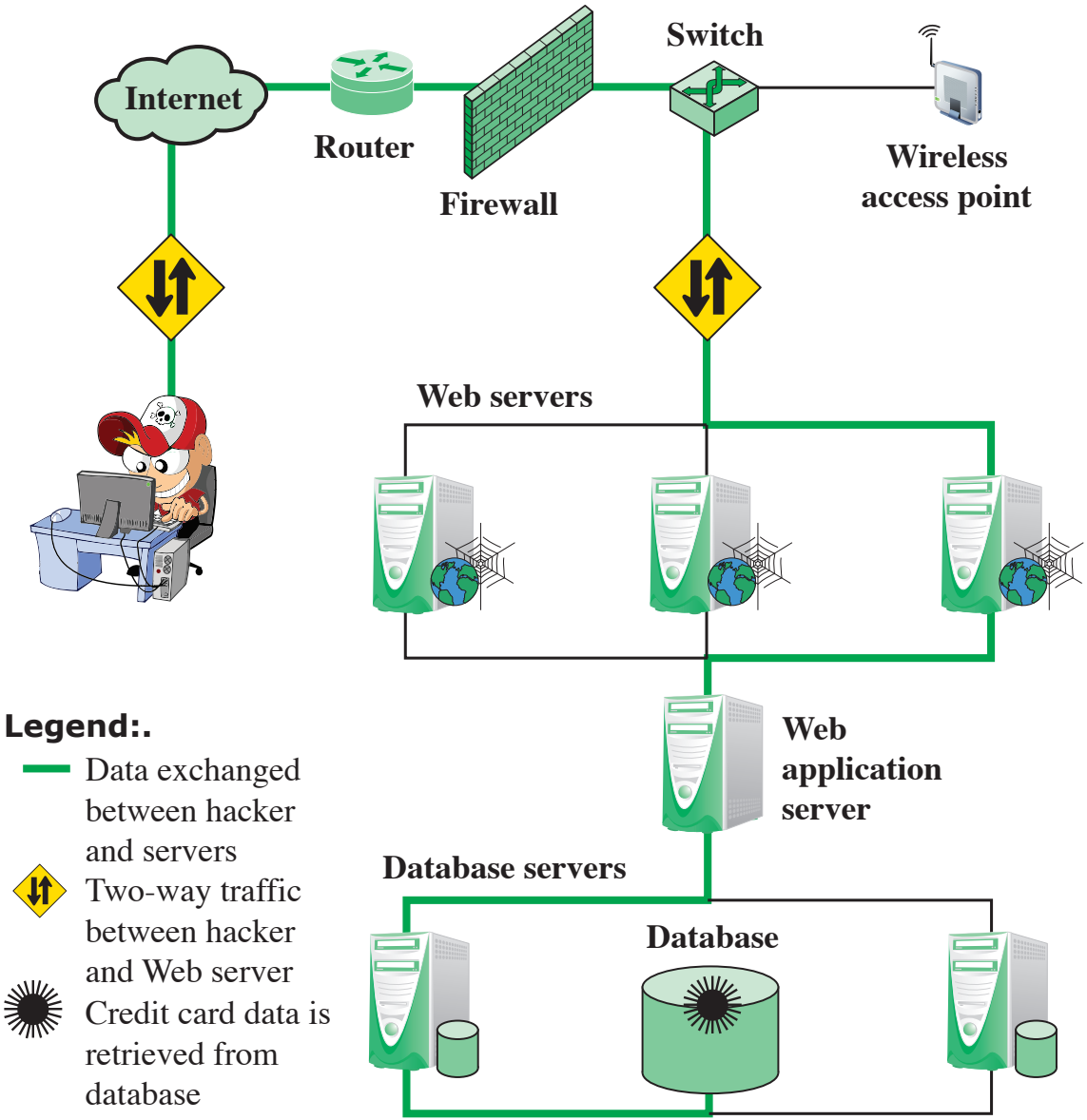


Figure 5.5 Typical SQL Injection Attack

Injection Technique

The SQLi attack typically works by prematurely terminating a text string and appending a new command

Because the inserted command may have additional strings appended to it before it is executed the attacker terminates the injected string with a comment mark “- -”

Subsequent text is ignored at execution time

Injection Techniques

var Shipcity;

ShipCity = Request.form (“ShipCity”);

var sql = “select * from OrdersTable where ShipCity = ‘ ” + ShipCity + “ ‘ ”

Intention: a user will enter the name of a city. When the script is executed, the user is prompted to enter a city, and if the user enters SF, then the following SQL query is generated

Injection Techniques

- SELECT * FROM OrderTable WHERE ShipCity = ‘SF’
- Suppose, however, the user enters the following
- ‘SF’; DROP table OrderTable --
- This results in the following SQL query:
- SELECT * FROM OrderTable WHERE ShipCity = ‘SF’; DROP table OrderTable--

Inband Attacks

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

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

Tautology

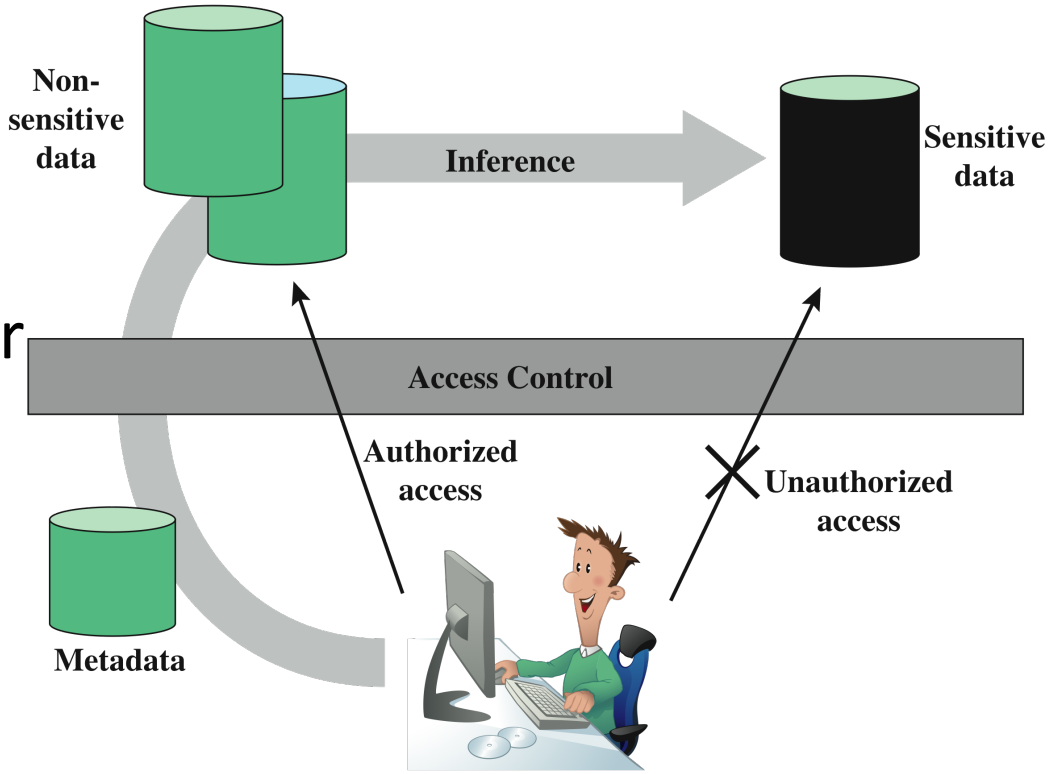
- Consider the following script whose intent is to require the user to enter a valid name and pwd
- $query = “SELECT info FROM user WHERE name = ‘$_GET [“name”]’ AND pwd = ‘$_GET [“pwd”]’ ”;
- Suppose the attacker submits “ ‘ ABC --’” for the name field. The resulting query will look like this
- SELECT info FROM users WHERE name = ‘ ‘ABC -- AND pwd = ’ ’
- The injected code effectively disables the pwd check (because of the comment indicator --)

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
- Include:
  - Illegal/logically incorrect queries
    - This attack lets an attacker gather important information about the type and structure of the backend database of a Web application
    - The attack is considered a preliminary, information-gathering step for other attacks
  - Blind SQL injection
    - Allows attackers to infer the data present in a database system even when the system is sufficiently secure to not display any erroneous information back to the attacker

Inference

- Performing queries to deduce unauthorized information from the legitimate responses received
- inference channel
  - information transfer path by which unauthorized data is obtained



Inference Example

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

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

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

Name	Department
Andy	strip
Calvin	strip
Cathy	strip

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

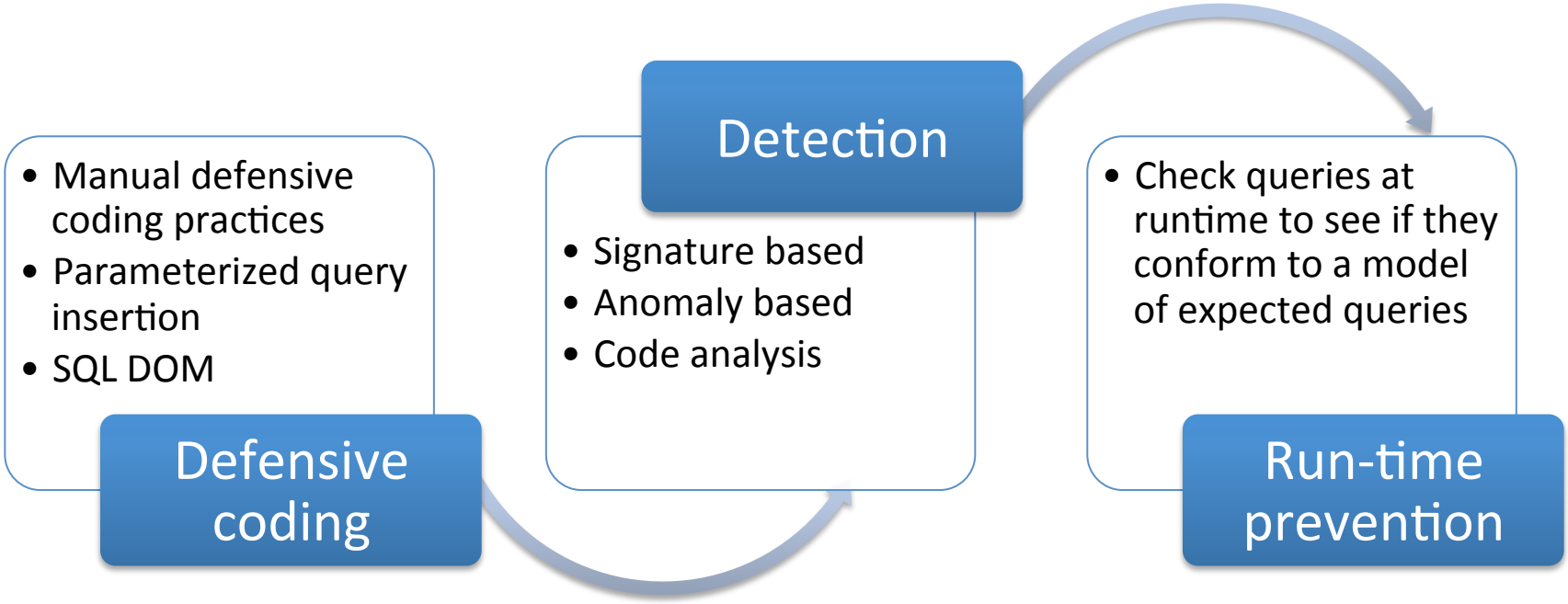
(b) Two views

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

(c) Table derived from combining query answers

SQLi Countermeasures

- Three types:



Database Access Control

database access control system determines:

if the user has access to the entire database or just portions of it

what access rights the user has

(create, insert, delete, update, read, write)

can support a range of administrative policies

centralized administration

- small number of privileged users may grant and revoke access rights

ownership-based administration

- the creator of a table may grant and revoke access rights to the table

decentralized administration

- 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
    - used to grant one or more access rights or can be used to assign a user to a role
  - revoke
    - revokes the access rights

```
GRANT { privileges | role }
[ON table]
TO { user | role | PUBLIC }
[IDENTIFIED BY password]
[WITH GRANT OPTION]
```

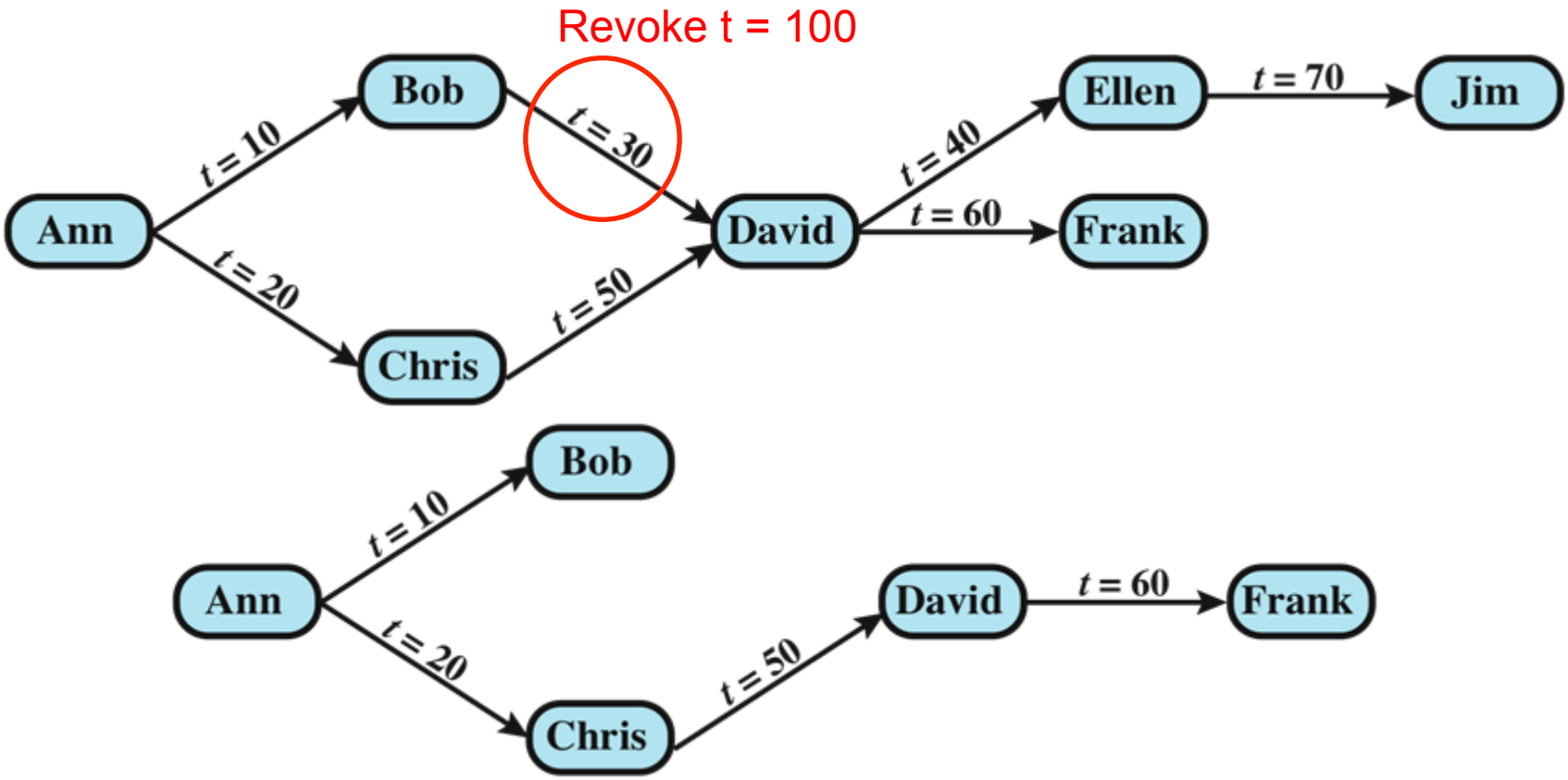
```
REVOKE { privileges | role }
[ON table]
FROM { user | role | PUBLIC }
```

SQL Access Controls

GRANT { privileges | role }
[ON table]
TO { user | role | PUBLIC }
[IDENTIFIED BY password]
[WITH GRANT OPTION]

- typical access rights are:
  - select, insert, update, delete, references

Cascading Authorizations

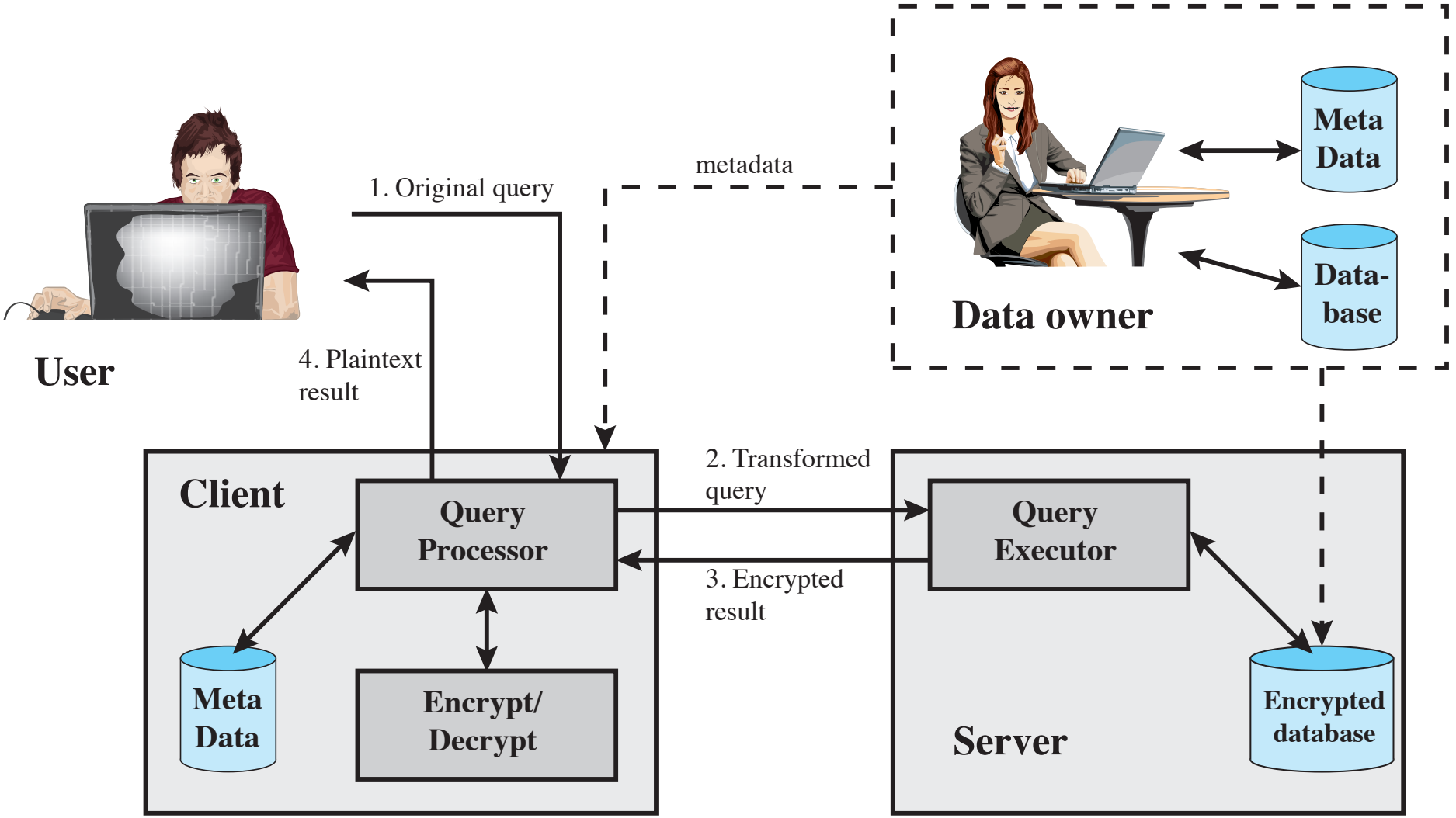


Rule: When user A revokes an access right, any cascaded access right is also revoked

Database Encryption

- database is typically the most valuable information resource for any organization
  - protected by multiple layers of security
    - firewalls, authentication, O/S access control systems, DB access control systems, database encryption
- encryption is often implemented with particularly sensitive data
  - at record, attribute, or individual field levels
- disadvantages to encryption:
  - key management
  - inflexibility

Database Encryption



Encryption Scheme

(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

Database Encryption

- Disadvantage of database encryption—inflexibility
- For example, a user wishes to retrieve all records for salaries less than $70K. There is no obvious way to do this, because the attribute value for salary in each record is encrypted (why?)
- One of current research topics in “searchable encrypted database”