LUND UNIVERSITY

EITA25 Computer Security (Datasäkerhet)
Databases

PAUL STANKOVSKI WAGNER, EIT, 2020-02-28

Database Security

- A database stores data but provides information
- Operating systems security
  - Controls access to data
  - File can be accessed based on identity and ACL
  - Content of file is not considered
- Database Security
  - Information is extracted from database
  - Access control also has to consider the content of the database

Database Security

- The sensitive information can include
  - Exact data – the values stored in the entries
  - Lower or upper bounds of values – highest and lowest salary can be sensitive
  - Negative result – the fact that an entry is not a particular value
  - Existence of data
  - Probable value
- Tradeoff between how much information to restrict and the value of having the database
- *Precision* – protect sensitive information while revealing as much nonsensitive information as possible
- *Consistency*
  - *Internal Consistency* – entries obey some predefined rules
  - *External Consistency* – entries are correct

Example, relational database

- Relations: Diary and Flights
- Attributes in relation Diary: Name, Day, Flight, Status
- Domain of attribute Day: {Mon, Tue, Wed, Thu, Fri, Sat, Sun}
- Tuple in Diary: {Bob, Mon, YL011, business)

Diary

Name	Day	Flight	Status
Alice	Mon	GR123	private
Bob	Mon	YL011	business
Bob	Wed	BX201	
Carol	Tue	BX201	business
Alice	Thu	FL9700	business

Flights

Flight	Destination	Departs	Days
GR123	THU	07:55	1-4-
YL011	ATL	08:10	12345-7
BX201	SLA	09:20	1-3-5-
FL9700	SLA	14:00	-2-4-6-
GR127	THU	14:55	-2-5-

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

SQL

- Structured Query Language – describes how information is retrieved and updated
- Data manipulation options: SELECT, UPDATE, INSERT, DELETE

Examples

```
SELECT Name, Status FROM Diary WHERE Day = 'Mon'

UPDATE Diary SET Status = private WHERE Day = 'Sun'

INSERT INTO Diary (Name, Day, Flight) VALUES ('Alice', 'Tue', 'BX201')

DELETE FROM Diary WHERE Name = 'Alice'
```

Using views

- A view is a virtual table usually selecting rows and columns from one or more base tables

Example

```
CREATE VIEW business_trips AS
        SELECT * FROM Diary WHERE Status = 'business'
```

- This will create a table with only business trips
- Views can be used for access control

Example

```
CREATE VIEW My_Journeys AS
        SELECT * FROM Diary WHERE Name = current_user()
```

- When a user queries this view, he will only see his own flights

Inference

- Derivation of sensitive information from nonsensitive data

Name	Position	Salary	Department	Manager
Alice	senior	45000	Finance	Alice
Bob	junior	30000	Finance	Alice
Charlie	senior	43000	Finance	Alice
Dennis	junior	29000	Technical	Fritz
Egon	senior	41000	Technical	Fritz
Fritz	senior	42000	Technical	Fritz

Employee

Access control prevents users from seeing table.
Only views with less sensitive information can be accessed.

CREATE VIEW view1 AS SELECT Position, Salary FROM Emplyee WHERE Department='Finance'

CREATE VIEW view2 AS SELECT Name, Department FROM Emplyee WHERE Department='Finance'

Position	Salary
senior	45000
junior	30000
senior	43000

view1

↔

Name	Department
Alice	Finance
Bob	Finance
Charlie	Finance

view2

If order is same, then
Name can be connected with Salary

Statistical Databases

- Only statistical queries are permitted (aggregation operators)
  - COUNT – The number of values in a column
  - SUM – The sum of the values in a column
  - AVG – The average of the values in a column
  - MAX – The largest value in a column
  - MIN – The smallest value in a column
- Problems
  - The individual data items are sensitive
  - Statistical queries read individual items
  - We want to provide users statistical information without compromising confidentiality of individual items

Statistical Database

Name	Sex	Program	Year	Credits	Average Grade
Alice	Female	C	18	60	3.4
Britta	Female	E	18	52	2.5
Carl	Male	E	16	63	3.5
Doris	Female	C	16	80	4.0
Ebbe	Male	D	17	50	2.2
Frank	Male	E	19	58	3.0
Gustav	Male	C	16	70	3.8
Helena	Female	F	17	58	2.8
Ingvar	Male	C	19	60	3.2
Julia	Female	D	17	75	3.8
Kajsa	Female	F	19	50	2.5
Lars	Male	E	16	60	3.0
Martin	Male	C	17	65	3.5

Students

Statistical queries are allowed, i.e., COUNT, SUM, AVG, MAX, MIN

We should not be able to see the grade of one individual → sensitive information

Trivial Attack

```
SELECT COUNT(*) FROM Students
   WHERE Sex = 'Female' and Program = 'D'
```
← Will return 1

```
SELECT AVG(Average Grade) FROM Students
   WHERE Sex = 'Female' and Program = 'D'
```
← Will return 3.8

- Problem when only a small subset is used
- Solution: require that at least *k* and at most *n-k* rows are represented in return calculation *(n* is total number of rows*)*
- However, let queries for all rows be ok.
- Not enough – Inference is still possible!

The General Tracker

- Let R uniquely identify row *r*
- Let T be a query predicate such that the set returned from T and the set returned from NOT(T) are large enough
- If we use both R OR T and R OR NOT(T) then *r* is the only row used in both queries
- T is called a **general tracker** (not tied to an individual)
  - T = (Sex='Female')

Example

Q1: SELECT SUM(Average Grade) FROM Students WHERE name = 'Julia' OR Sex = 'Female'

Q2: SELECT SUM(Average Grade) FROM Students WHERE name = 'Julia' OR NOT(Sex = 'Female')

Q3: SELECT SUM(Average Grade) FROM Students

Julia's Average Grade given by Q1 + Q2 – Q3

General Tracker, Example

Name	Sex	Program	Year	Credits	Average Grade
Alice	Female	C	08	60	3.4
Britta	Female	E	08	52	2.5
Carl	Male	E	06	63	3.5
Doris	Female	C	06	80	4.0
Ebbe	Male	D	07	50	2.2
Frank	Male	E	09	58	3.0
Gustav	Male	C	06	70	3.8
Helena	Female	F	07	58	2.8
Ingvar	Male	C	09	60	3.2
Julia	Female	D	07	75	3.8
Kajsa	Female	F	09	50	2.5
Lars	Male	E	06	60	3.0
Martin	Male	C	07	65	3.5

Q1: Sum = 19.0

Name	Sex	Program	Year	Credits	Average Grade
Alice	Female	C	08	60	3.4
Britta	Female	E	08	52	2.5
Carl	Male	E	06	63	3.5
Doris	Female	C	06	80	4.0
Ebbe	Male	D	07	50	2.2
Frank	Male	E	09	58	3.0
Gustav	Male	C	06	70	3.8
Helena	Female	F	07	58	2.8
Ingvar	Male	C	09	60	3.2
Julia	Female	D	07	75	3.8
Kajsa	Female	F	09	50	2.5
Lars	Male	E	06	60	3.0
Martin	Male	C	07	65	3.5

Q2: Sum = 26.0

Q3: Sum = 41.2

Julias grade = 19.0 + 26.0 - 41.2 = 3.8

Differential Privacy (DP)

Idea: Use noise to distort data

Cynthia Dwork, Frank McSherry, Kobbi Nissim and Adam Smith:

Calibrating Noise to Sensitivity in Private Data Analysis

Theory of Cryptography Conference (TCC) 2006 (DOI:10.1007/11681878 14),
Full version in Journal of Privacy and Confidentiality (DOI:10.29012/jpc.v7i3.405).

Gödel prize in 2017 for outstanding paper in theoretical computer science.

Differential Privacy (DP)

Dwork et al. define ε-differential privacy as

> A randomized function K gives ε-differential privacy if for all data sets D1 and D2 differing on at most one element, and all S ⊆ Range(K), $\Pr[K(D1) \in S] \leq \exp(\varepsilon) \times \Pr[K(D2) \in S]$.

Parameter ε can be thought of as an overall privacy protection against DP analysis.

- lower value means better protection
- ε = 0 means perfect privacy (useless since all input data must be ignored)

” In many cases, extremely accurate information about the database can be provided while simultaneously ensuring very high levels of privacy.”

Differential Privacy (DP)

> ” In many cases, extremely accurate information about the database can be provided while simultaneously ensuring very high levels of privacy.”

- Amount of information leakage from a single database query depends on
  - how much of my information was used
  - how much other information was used

- Solution: adjust noise level in output to achieve *sufficient* level of privacy
  - assign ε according to your definition of sufficient
  - use more or less noise adaptively
    - » more if fewer users contribute to output
    - » less if many users contribute

I though I would bring some noise!

LUND
UNIVERSITY