Outline

• Security in Relational Database Systems
• Security in Statistical Databases
• Current Trends

Discretionary Access Control in SQL

GRANT privileges ON object TO users [WITH GRANT OPTIONS]

privileges = SELECT | INSERT(column-name) | DELETE | REFERENCES(column-name)
object = table | attribute

Examples

GRANT INSERT, DELETE ON Reserves TO Yuppy WITH GRANT OPTIONS
GRANT SELECT ON Reserves TO Michael
GRANT SELECT ON Sailors TO Michael WITH GRANT OPTIONS
GRANT UPDATE (rating) ON Sailors TO Leah
GRANT REFERENCES (bid) ON Boats TO Bill

Views and Security

• David has SELECT rights on table Students
• Creates a VIEW BrightStudents
• Grants SELECT rights on BrightStudents to Dan

Revocation

REVOKE [GRANT OPTION FOR] privileges
ON object FROM users { RESTRICT | CASCADE }

Administrator says:

REVOKE SELECT ON Students FROM David CASCADE

Dan loses SELECT privileges on BrightStudents
Revocation

Joe: GRANT [...] TO Art ...
Art: GRANT [...] TO Bob ...
Bob: GRANT [...] TO Art ...
Joe: GRANT [...] TO Cal ...
Cal: GRANT [...] TO Bob ...
Joe: REVOKE [...] FROM Art CASCADE

What happens ??

According to SQL everyone keeps the privilege

Attacks

• SQL injection (in class)

Security in Statistical Databases

Goal:
• Allow aggregate queries
• Hide confidential data

Why it’s hard:
• Allow arbitrary aggregate queries, as long as no compromise

Table

<table>
<thead>
<tr>
<th>Age</th>
<th>Sex</th>
<th>Employer</th>
<th>Diagnosis</th>
</tr>
</thead>
<tbody>
<tr>
<td>42</td>
<td>M</td>
<td>ABC</td>
<td>Schizophrenia</td>
</tr>
<tr>
<td>25</td>
<td>F</td>
<td>XYZ</td>
<td>Depression</td>
</tr>
<tr>
<td>42</td>
<td>F</td>
<td>XYZ</td>
<td>Depression</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Security in Statistical Databases

Queries

SELECT count(*)
FROM Table
WHERE Age=42 and Sex='M' and Employer='ABC'

Allow arbitrary conditions
Attacks

• Mallory knows about John Smith:
  \[\text{Age}=42 \land \text{Sex}='M' \land \text{Employer}='ABC'\]

• Query 1:
  \[\text{count(Age}=42 \land \text{Sex}='M' \land \text{Employer}='ABC')\]
  Answer= 1  we’re lucky!

• Query 2:
  \[\text{count(Age}=42 \land \text{Sex}='M' \land \text{Employer}='ABC' \land \text{Diagnosis} = \text{'Schizophrenia'}\]
Hippocratic Databases

- For the pragmatists
- IBM Almaden [Agrawal et al.]
- Hippocratic Oath: “...I will remain silent...”
- Hippocratic Databases: ten principles:
  - Purpose specification
  - Consent
  - Limited collection
  - Limited use
  - etc

Security in Data Exchange

- Secrecy: make sure you don’t give away data when you don’t mean to
- Integrity: how can you verify that the data you download is unchanged from its original form?

Latanya Sweeney’s Finding

- In Massachusetts, the Group Insurance Commission (GIC) is responsible for purchasing health insurance for state employees
- GIC collects data, and since it’s “private”, it publishes it:

\[
\text{GIC}(\text{zip, dob, sex, diagnosis, procedure, ...})
\]

\[
\text{VOTER(name, party, ..., zip, dob, sex)}
\]

Latanya Sweeney’s Finding

- William Weld (former governor) lives in Cambridge, hence is in VOTER
- 6 people in VOTER share his dob
- only 3 of them were man (same sex)
- Weld was the only one in that zip
- Sweeney learned Weld’s medical records!

Current proposed solution: k-anonymity

Secrecy in Data Exchange

- Enforce access control policies with encryption
- Start with the plain XML document, then encrypt all fragments that need to stay secret
- Only users having the right key have access
- Problem: multiple policies
Secrecy in Data Exchange

XML File:

```
医院
患者
患者
患者
患者
姓名
楼层
疾病
```

Smith 3 Schizophrenia

Secrecy in Data Exchange

Mr. Smith’s disease is accessible to:
• Physicians
• Nurses working on the 3rd floor

Keys: $K_{\text{physician}}, K_{\text{nurse}}, K_{\text{3rd}}$

How do we encrypt? Need $K_{\text{physician}} \lor K_{\text{nurse}} \land K_{\text{3rd}}$

Secrecy in Data Exchange

```
医院
患者
患者
患者
患者
姓名
楼层
疾病
```

There is a standard for that…

Secrecy in Data Exchange

```
医院
患者
患者
患者
患者
姓名
楼层
疾病
```

$K_{\text{physician}}, K_{\text{nurse}}, K_{\text{3rd}}$

$s = s_1 \text{ XOR } s_2$

$s_1, s_2$

Schizophrenia

Secure Information Sharing

• Agrawal, Efvimievski, Srikant [SIGMOD’2003]

• Example: two competing companies agree to share their list of their customers with a poor payment record but nothing else

Secure Information Sharing

Formally:

• Alice has $A = \{x_1, \ldots, x_n\}$
• Bob has $B = \{y_1, \ldots, y_m\}$

• They want to find out $A \cap B$, and not reveal anything else
Secure Information Sharing

Attempt 1:
• Alice computes $H_A = h(A)$ sends to Bob
• Bob computes $H_B = h(B)$ sends to Alice
• Now each computes $A \cap B$
• What’s wrong?

Solution: use commutative encryption

Example: $E_k(x) = x^k \mod p$

Secure Information Sharing

Solution:
1. Alice computes $Y_A = \{E_a(x) \mid x \in A\}$
2. Bob computes $Y_B = \{E_b(y) \mid y \in B\}$
3. Exchange $Y_A, Y_B$ ORDERED!
4. Bob computes $\{(E_a(x), E_b(E_a(x))) \mid x \in A\}$
5. Alice computes $\{E_a(E_b(y)) \mid y \in B\}$
6. Bob sends that to Alice
7. Alice can now compute $A \cap B$

Integrity in Data Sharing

• Merkle Trees (in class)