Database Security

CS 645

Apr 15, 2008

Gerome Miklau, UMass Amherst
Last time

- Security basics
- Access control in Databases
- Privacy foundations
Today

- Beyond access control
  - What do views really protect?
  - Statistical DBs and data perturbation
  - SQL injection
  - Overview of other DB security research
Limitations of access control

• Trusted enforcement required
• Balancing protection and sharing of data
• Controlling direct access != protecting information
  • Unexpected disclosures
  • Inferences
  • Background knowledge
Object of protection: the logical view

- Views can protect sensitive data or relationships

database $X$  
View $V$  
View Result $V(X)$

<table>
<thead>
<tr>
<th>name</th>
<th>age</th>
<th>salary</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

$V$ (name, age, salary) for Employees over 40
Publish v. Protect

Two competing goals

1. Share data and permit collaboration
   - Publish a view $V$
   - “Names and ages of employees”

2. Protect sensitive items & relationships
   - Protect a query $S$
   - “Name and salary association”

Understanding disclosure:
What is the relationship between $V$ and $S$?
Answerability

Database X

View V

View Result V(X)

<table>
<thead>
<tr>
<th>name</th>
<th>age</th>
<th>salary</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

V: name, age for Employees over 40

<table>
<thead>
<tr>
<th>name</th>
<th>age</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

S: name, age for Employees over 50

name, age

?
Shift of perspective

Concern: data availability

When/how can I use a published view to answer my query?

Traditional results: answerability
A priori knowledge

Hidden database $X$ is one of these

Sensitive answer is one of these:

All possible databases $X$  All possible answers $S(X)$
Unknown database $X$ is one of these

Not all databases consistent with $V(X)$

Not all possible answers to $S(X)$ consistent with $V(X)$
Probabilistic knowledge

Knowledge about S, *a priori* 

- Schema
- Domain
- Probability of each database

Knowledge about S, given V

+ view definition V
+ view result V(X)

Probability of each answer to S

Probability of each answer to S
Definition: query-view security

S and V are secure if:

\[ \Pr[S(X)=u] = \Pr[S(X)=u \mid V(X)=v] \quad \forall u \forall v \]

- a priori knowledge
- knowledge given V
- must hold for all answers

• Inspired by perfect secrecy [Shannon 1949]
• Independence of probabilistic events
Concrete example

Suppose the database represents a directed graph over domain of 2 nodes A and B.

<table>
<thead>
<tr>
<th>source</th>
<th>destination</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>B</td>
</tr>
<tr>
<td>B</td>
<td>B</td>
</tr>
</tbody>
</table>

\[ S = \{ B \} \]
\[ V = \{ A, B \} \]
16 possible graphs (equally likely)

S nodes that are a destination = ?

<table>
<thead>
<tr>
<th>answer to S</th>
<th>a priori probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>{}</td>
<td>1/16</td>
</tr>
<tr>
<td>{A}</td>
<td>3/16</td>
</tr>
<tr>
<td>{B}</td>
<td>3/16</td>
</tr>
<tr>
<td>{A, B}</td>
<td>9/16</td>
</tr>
</tbody>
</table>

Published view:

V nodes that are a source = \{A\}

Conclusion: \(V\) and \(S\) not secure.
Goal: logical criterion

Brute force $\rightarrow$ Logical criterion

For each answer to $S$ answer to $V$
Compare probabilities

Analyze the view/query expressions $V$ and $S$

(depends on domain & probability distribution)
Main results

<table>
<thead>
<tr>
<th>Decision procedure</th>
<th>Given views { ( V_1 \ldots V_n ) } and query ( S ), decide whether security holds.</th>
</tr>
</thead>
</table>

Language: conjunctive queries with inequalities

Probabilities: tuple-independent

[M., Suciu. SIGMOD 2004]
Main insight

When does a view or query expression “contain information” about a tuple?

Overlap in critical tuples acts as an information channel.
Critical tuples

A tuple t is critical for a query Q if there is a database instance X containing t such that:

$$Q(X) \neq Q(X - \{t\})$$

for example:

<table>
<thead>
<tr>
<th>source</th>
<th>dest</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>B</td>
</tr>
<tr>
<td>B</td>
<td>B</td>
</tr>
</tbody>
</table>

V and S secure if and only if

V(X) is \{A, B\}

X - \{t\} is disjoint

<table>
<thead>
<tr>
<th>source</th>
<th>dest</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>B</td>
</tr>
</tbody>
</table>

crit(V) and crit(S) disjoint

V(X - \{t\}) is \{B\}
### Main results

<table>
<thead>
<tr>
<th>Decision procedure</th>
<th>Given views ${V_1 \ldots V_n}$ and query $S$, decide whether security holds.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Complexity</td>
<td>Deciding security is $\Pi_2^P$-complete.</td>
</tr>
<tr>
<td>Robustness</td>
<td>Security is independent of domain and probability distribution.</td>
</tr>
</tbody>
</table>

Language: conjunctive queries with inequalities
Probabilities: tuple-independent

[Miklau, Suciu. SIGMOD 2004]
Spectrum of disclosure

Employee (name, dept, salary)

Published View | Sensitive Query | Disclosure
---|---|---
$V_1(\text{name, salary})$ | $S_1(\text{name})$ | total

where dept="admin"

$S_4(\text{name, salary})$ where dept="sales" none
Related & subsequent work

- **Benefits**
  - captures subtle disclosures
  - logical characterization
  - models collusion, outside knowledge, FDs

- **Limitations**
  - some acceptable disclosures rejected
  - high complexity

- **Extensions, subsequent results**
  - [Dalvi, M., Suciu. ICDT 2005]
  - [Deutsch, Papakonstaninou. ICDT 2005]
  - [Machanavajjhala, Gehrke. PODS 2006]
Today

- Beyond access control
  - What do views really protect?
  - **Statistical DBs and data perturbation**
  - SQL injection
  - Overview of other DB security research
Inference in Statistical DBs

- **Statistical database**
  - A database which permits users to retrieve aggregate statistics (e.g. count, average) only for subsets of entities.

- **Goal:**
  - share aggregate data
  - avoid exact or partial disclosures of entity attributes.

- **Common applications:** census or medical data
Statistical databases

SELECT count(*)
FROM Patients
WHERE age=42
    and sex=‘M’
    and diagnostic=‘schizophrenia’

SELECT name
FROM Patient
WHERE age=42
    and sex=‘M’
    and diagnostic=‘schizophrenia’
Solution approaches

Query restriction

(Restricted) queries

Data perturbation

Perturbed SDB

Output perturbation

Perturbed responses
“Anonymized” data publishing

- Mass. Group Insurance Commission (GIC) is responsible for purchasing health insurance for state employees
- GIC collects data, and publishes it:

<table>
<thead>
<tr>
<th>name</th>
<th>ssn</th>
<th>gender</th>
<th>dob</th>
<th>zip</th>
<th>diagnosis</th>
</tr>
</thead>
<tbody>
<tr>
<td>joe</td>
<td>123-56-7634</td>
<td>male</td>
<td>1/4/64</td>
<td>1045</td>
<td>cancer</td>
</tr>
<tr>
<td>mary</td>
<td>113-36-4252</td>
<td>female</td>
<td>3/24/45</td>
<td>1312</td>
<td>flu</td>
</tr>
<tr>
<td>bob</td>
<td>124-46-1574</td>
<td>male</td>
<td>5/4/55</td>
<td>1452</td>
<td>HIV</td>
</tr>
</tbody>
</table>

Identifier Sensitive Attr.
Additional data source

- Sweeney paid $20 and bought the public voter registration list for Cambridge Mass.:

<table>
<thead>
<tr>
<th>name</th>
<th>party</th>
<th>gender</th>
<th>dob</th>
<th>zip</th>
</tr>
</thead>
<tbody>
<tr>
<td>joe</td>
<td>Dem</td>
<td>male</td>
<td>1/4/64</td>
<td>1045</td>
</tr>
<tr>
<td>mary</td>
<td>Rep</td>
<td>female</td>
<td>3/24/45</td>
<td>1312</td>
</tr>
<tr>
<td>bob</td>
<td>Dem</td>
<td>male</td>
<td>5/4/55</td>
<td>1452</td>
</tr>
</tbody>
</table>
Re-identification

INSURED(zip, dob, sex, diagnosis, procedure, ...)
VOTER(name, party, ..., zip, dob, sex)

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

Latanya Sweeney’s Finding

87% of the US population (216 million out of 248 million) are likely to be uniquely identified based on:

zipcode, gender, date-of-birth
K-Anonymity

- Intuition: privacy is gained by hiding individuals in groups of sufficient size

- Alter data so that:
  - At least k individuals share each pseudo-identifier occurring in the database.
  - Attribute **suppression** and **generalization**

**Data perturbation**

![Diagram showing data perturbation process](image-url)
## K-anonymity example

### Original data

<table>
<thead>
<tr>
<th>Non-Sensitive</th>
<th>Sensitive</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zip Code</td>
<td>Age</td>
</tr>
<tr>
<td>1</td>
<td>13053</td>
</tr>
<tr>
<td>2</td>
<td>13068</td>
</tr>
<tr>
<td>3</td>
<td>13068</td>
</tr>
<tr>
<td>4</td>
<td>13053</td>
</tr>
<tr>
<td>5</td>
<td>14853</td>
</tr>
<tr>
<td>6</td>
<td>14853</td>
</tr>
<tr>
<td>7</td>
<td>14850</td>
</tr>
<tr>
<td>8</td>
<td>14850</td>
</tr>
<tr>
<td>9</td>
<td>13053</td>
</tr>
<tr>
<td>10</td>
<td>13053</td>
</tr>
<tr>
<td>11</td>
<td>13068</td>
</tr>
<tr>
<td>12</td>
<td>13068</td>
</tr>
</tbody>
</table>

### 4-anonymous data

<table>
<thead>
<tr>
<th>Non-Sensitive</th>
<th>Sensitive</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zip Code</td>
<td>Age</td>
</tr>
<tr>
<td>1</td>
<td>130**</td>
</tr>
<tr>
<td>2</td>
<td>130**</td>
</tr>
<tr>
<td>3</td>
<td>130**</td>
</tr>
<tr>
<td>4</td>
<td>130**</td>
</tr>
<tr>
<td>5</td>
<td>1485*</td>
</tr>
<tr>
<td>6</td>
<td>1485*</td>
</tr>
<tr>
<td>7</td>
<td>1485*</td>
</tr>
<tr>
<td>8</td>
<td>1485*</td>
</tr>
<tr>
<td>9</td>
<td>130**</td>
</tr>
<tr>
<td>10</td>
<td>130**</td>
</tr>
<tr>
<td>11</td>
<td>130**</td>
</tr>
<tr>
<td>12</td>
<td>130**</td>
</tr>
</tbody>
</table>

From Machanavajjhala, Gehrke, Kifer. ICDE 06
Analysis

- Higher k -- more privacy
- Fewer suppressions & generalizations -- more accuracy

<table>
<thead>
<tr>
<th>Zip Code</th>
<th>Age</th>
<th>Nationality</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>130**</td>
<td>&lt; 30</td>
<td>*</td>
</tr>
<tr>
<td>2</td>
<td>130**</td>
<td>&lt; 30</td>
<td>*</td>
</tr>
<tr>
<td>3</td>
<td>130**</td>
<td>&lt; 30</td>
<td>*</td>
</tr>
<tr>
<td>4</td>
<td>130**</td>
<td>&lt; 30</td>
<td>*</td>
</tr>
<tr>
<td>5</td>
<td>1485*</td>
<td>≥ 40</td>
<td>*</td>
</tr>
<tr>
<td>6</td>
<td>1485*</td>
<td>≥ 40</td>
<td>*</td>
</tr>
<tr>
<td>7</td>
<td>1485*</td>
<td>≥ 40</td>
<td>*</td>
</tr>
<tr>
<td>8</td>
<td>1485*</td>
<td>≥ 40</td>
<td>*</td>
</tr>
<tr>
<td>9</td>
<td>130**</td>
<td>3*</td>
<td>*</td>
</tr>
<tr>
<td>10</td>
<td>130**</td>
<td>3*</td>
<td>*</td>
</tr>
<tr>
<td>11</td>
<td>130**</td>
<td>3*</td>
<td>*</td>
</tr>
<tr>
<td>12</td>
<td>130**</td>
<td>3*</td>
<td>*</td>
</tr>
</tbody>
</table>
Attacks on anonymized data

<table>
<thead>
<tr>
<th>Zip Code</th>
<th>Age</th>
<th>Nationality</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>130**</td>
<td>&lt; 30</td>
<td>*</td>
<td>Heart Disease</td>
</tr>
<tr>
<td>130**</td>
<td>&lt; 30</td>
<td>*</td>
<td>Heart Disease</td>
</tr>
<tr>
<td>130**</td>
<td>&lt; 30</td>
<td>*</td>
<td>Viral Infection</td>
</tr>
<tr>
<td>130**</td>
<td>&lt; 30</td>
<td>*</td>
<td>Viral Infection</td>
</tr>
<tr>
<td>1485*</td>
<td>≥ 40</td>
<td>*</td>
<td>Cancer</td>
</tr>
<tr>
<td>1485*</td>
<td>≥ 40</td>
<td>*</td>
<td>Heart Disease</td>
</tr>
<tr>
<td>1485*</td>
<td>≥ 40</td>
<td>*</td>
<td>Viral Infection</td>
</tr>
<tr>
<td>1485*</td>
<td>≥ 40</td>
<td>*</td>
<td>Viral Infection</td>
</tr>
<tr>
<td>130**</td>
<td>3*</td>
<td>*</td>
<td>Cancer</td>
</tr>
<tr>
<td>130**</td>
<td>3*</td>
<td>*</td>
<td>Cancer</td>
</tr>
<tr>
<td>130**</td>
<td>3*</td>
<td>*</td>
<td>Cancer</td>
</tr>
<tr>
<td>130**</td>
<td>3*</td>
<td>*</td>
<td>Cancer</td>
</tr>
</tbody>
</table>

- **Homogeneity attack**
  - Alice knows Bob is 31, living in zip 13053

- **Background knowledge attack**
  - Alice has Japanese friend who is 21 and living in 13068.
### Issues

A 4-anonymous, 3-diverse table

<table>
<thead>
<tr>
<th>Zip Code</th>
<th>Age</th>
<th>Nationality</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1305*</td>
<td>≤ 40</td>
<td>Heart Disease</td>
</tr>
<tr>
<td>4</td>
<td>1305*</td>
<td>≤ 40</td>
<td>Viral Infection</td>
</tr>
<tr>
<td>9</td>
<td>1305*</td>
<td>≤ 40</td>
<td>Cancer</td>
</tr>
<tr>
<td>10</td>
<td>1305*</td>
<td>≤ 40</td>
<td>Cancer</td>
</tr>
<tr>
<td>5</td>
<td>1485*</td>
<td>&gt; 40</td>
<td>Cancer</td>
</tr>
<tr>
<td>6</td>
<td>1485*</td>
<td>&gt; 40</td>
<td>Heart Disease</td>
</tr>
<tr>
<td>7</td>
<td>1485*</td>
<td>&gt; 40</td>
<td>Viral Infection</td>
</tr>
<tr>
<td>8</td>
<td>1485*</td>
<td>&gt; 40</td>
<td>Viral Infection</td>
</tr>
<tr>
<td>2</td>
<td>1306*</td>
<td>≤ 40</td>
<td>Heart Disease</td>
</tr>
<tr>
<td>3</td>
<td>1306*</td>
<td>≤ 40</td>
<td>Viral Infection</td>
</tr>
<tr>
<td>11</td>
<td>1306*</td>
<td>≤ 40</td>
<td>Cancer</td>
</tr>
<tr>
<td>12</td>
<td>1306*</td>
<td>≤ 40</td>
<td>Cancer</td>
</tr>
</tbody>
</table>
Quality metrics

- **Bias**
  - difference between unperturbed statistic and expected value of perturbed output.

- **Precision**
  - variance of statistic

- **Consistency**
  - Answers consistent across queries
Supplemental knowledge

• In controlling inference, must model the adversary’s supplemental knowledge:
  • all information about the database the adversary can acquire from sources other than the system itself.
Today

- Beyond access control
  - What do views really protect?
  - Statistical DBs and data perturbation
- SQL injection
- Overview of other DB security research
SQL Injection

- Popular attack on databases accessed through web interfaces.
- Attacker is able to insert SQL statements into a query by manipulating application input data.
- Ranked as a top 10 security vulnerability
  - SANS Institute
  - Open Web Application Security Project (OWASP)

<table>
<thead>
<tr>
<th></th>
<th>Top Vulnerabilities in Web Applications</th>
</tr>
</thead>
<tbody>
<tr>
<td>A1</td>
<td>Unvalidated Input</td>
</tr>
<tr>
<td></td>
<td>Information from web requests is not validated before being used by a web application. Attackers can use these flaws to attack backend components through a web application.</td>
</tr>
<tr>
<td>A2</td>
<td>Broken Access Control</td>
</tr>
<tr>
<td></td>
<td>Restrictions on what authenticated users are allowed to do are not properly enforced. Attackers can exploit these flaws to access other users' accounts, view sensitive files, or use unauthorized functions.</td>
</tr>
</tbody>
</table>
### SQL injection: example

#### Treatment table

<table>
<thead>
<tr>
<th>patient</th>
<th>doctor</th>
<th>date</th>
<th>diagnosis</th>
</tr>
</thead>
<tbody>
<tr>
<td>fred</td>
<td>Dr. Lee</td>
<td>9/1/2005</td>
<td>cancer</td>
</tr>
<tr>
<td>mary</td>
<td>Dr. Lee</td>
<td>5/2/2004</td>
<td>flu</td>
</tr>
<tr>
<td>fred</td>
<td>Dr. Ash</td>
<td>1/18/2005</td>
<td>diabetes</td>
</tr>
<tr>
<td>joe</td>
<td>Dr. Boul</td>
<td>6/4/2005</td>
<td>flu</td>
</tr>
</tbody>
</table>
SQL Injection

Your health insurance company has a web site for claims:

First login:

- User: fred
- Password: ********

Then search:

- Search claims: Dr. Lee

SELECT...FROM...WHERE doctor='Dr. Lee' and patient='fred'

[Chris Anley, Advanced SQL Injection In SQL]
SQL Injection

Now try this:

Search claims: Dr. Lee’ OR patientID = ‘mary’;  --

.....WHERE doctor=‘Dr. Lee’ OR patientID=‘mary’;  --’ and patientID=‘fred’

Even better:

Search claims: Dr. Lee’ OR 1 = 1;  --
SQL Injection

- Those attacks threaten confidentiality
- There are also authenticity attacks
  - these often require knowledge of schema
  - can be discovered using error messages!

Extreme case:

Search claims: Dr. Lee'; DROP TABLE Patients; --
Determining schema

Illegal GROUP BY can reveal column names

Microsoft OLE DB Provider for ODBC Drivers error '80040e14'
[Microsoft][ODBC SQL Server Driver][SQL Server]Column 'users.username' is invalid in the select list because it is not contained in either an aggregate function or the GROUP BY clause.

Illegal UNION can reveal data types

Username: ' union select sum(username) from users--'

Microsoft OLE DB Provider for ODBC Drivers error '80040e07'
[Microsoft][ODBC SQL Server Driver][SQL Server]The sum or average aggregate operation cannot take a varchar data type as an argument.
Solutions

• Input validation
  • check content, length, format
  • generally a pain, hard to check, may accidentally reject good input

• Stored procedures with parameters

• Deeper answer:
  • Move policy implementation from apps to DB
HI, THIS IS YOUR SON'S SCHOOL. WE'RE HAVING SOME COMPUTER TROUBLE.

OH, DEAR — DID HE BREAK SOMETHING?

IN A WAY——

DID YOU REALLY NAME YOUR SON Robert'); DROP TABLE Students;-- ?

OH, YES. LITTLE BOBBY TABLES, WE CALL HIM.

WELL, WE'VE LOST THIS YEAR'S STUDENT RECORDS. I HOPE YOU'RE HAPPY.

AND I HOPE YOU'VE LEARNED TO SANITIZE YOUR DATABASE INPUTS.
Today

• Beyond access control
• What do views really protect?
• Statistical DBs and data perturbation
• SQL injection
• Overview of other DB security research
Other problems in Secure Data Management

- Forensics and auditing
- Multi-party secure query processing
- Private query processing
- Rights Management (watermarking)
- Database service model
Computer forensics

- Computer systems leave unintended remnants of data and past activities.
- Forensics is concerned with collection and analysis.
  - Good: legitimate investigations
  - Bad: unauthorized gathering of private information.
- How much data does a database retain, and for how long?
- How can we built a forensically “clean” system?
Auditing v. Retention policies

- Historical retention of data is an important aspect of privacy
- Retention periods limited by regulations, fear of release
- A historical record of past data and operations enables **auditing**
- How can we audit **and** obey limited retention policies?
Multi-party secure querying

- Alice has a database DB_A
- Bob has a database DB_B

- Goal: compute $Q(DB_A, DB_B)$, without revealing their data?
  - Bob should not see all of DB_A
  - Alice should not see all of DB_B
  - Both are willing to share $Q(DB_A, DB_B)$
**Example**

**Alice:** I am teaching Databases, and I suspect I have a cheater in my class

**Bob:** I am teaching Security, and I also suspect a cheater!

**Alice:** Tell me your suspect’s name! If it’s the same as mine, then I’ll know for sure he is cheating.

**Bob:** No. I’m not sure my suspect is cheating. Tell me your suspect’s name first, and if they match I’ll let you know.

**Alice:** No.
Private Query Processing

• Scenario: a service offers access to a useful database to customers

• Goal: want to process user queries, but keep queries secret from the engine

• Special case: “Private Information Retrieval”, PIR
Watermarking

- Alice wants to sell relational data
- But wants to detect unauthorized duplication
- Watermark:
  - small, hidden perturbations in the database; evidence of its origin
- Challenge
  - must withstand removal of many rows and/or columns
Database as a Service

Scenario:

- Alice has a database D
- Bob has a database engine, and offers to store and process D, for a fee
- But Alice doesn’t trust Bob, and wants to hide the data from him