kb-Anonymity: A Model for Anonymized Behavior-Preserving Test and Debugging Data

Aditya Budi, David Lo, Lingxiao Jiang, Lucia
Software Testing & Debugging

- Programs may fail
  - In-house during development process
  - Post-deployment in user fields
Where Come Inputs for Testing & Debugging?

- In-house generation
Where Come Inputs for Testing & Debugging?

- From clients
However, Privacy!

- From clients

Privacy Concerns!
Sample Privacy Leak

- Linking attack

Patient Records (private)

<table>
<thead>
<tr>
<th>Gender</th>
<th>Zipcode</th>
<th>DOB</th>
<th>Disease</th>
</tr>
</thead>
<tbody>
<tr>
<td>Male</td>
<td>95110</td>
<td>6/7/72</td>
<td>Heart Disease</td>
</tr>
<tr>
<td>Female</td>
<td>95110</td>
<td>1/31/80</td>
<td>Hepatitis</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Voter Registration List (public)

<table>
<thead>
<tr>
<th>Name</th>
<th>DOB</th>
<th>Gender</th>
<th>Zipcode</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bob</td>
<td>6/7/72</td>
<td>Male</td>
<td>95110</td>
</tr>
<tr>
<td>Beth</td>
<td>1/31/80</td>
<td>Female</td>
<td>95110</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Bob has heart disease
Sample Privacy Leak

- **Linking attack**

  - **Quasi-identifier fields**

    - **Patient Records (private)**
      - Gender | Zipcode | DOB      | Disease
      - Male   | 95110   | 6/7/72   | Heart Disease
      - Female | 95110   | 1/31/80  | Hepatitis
      - ...    | ...     | ...      | ...

    - **Voter Registration List (public)**
      - Name | DOB      | Gender | Zipcode
      - Bob  | 6/7/72   | Male   | 95110
      - Beth | 1/31/80  | Female | 95110
      - ...  | ...      | ...    | ...

  - **Bob has heart disease**

  - **Patient Records (private)**
    - Gender | Zipcode | DOB | Disease
    - Male   | *      | *   | Heart Disease
    - Female | *      | *   | Hepatitis
    - ...    | ...    | ... | ...

- **kb-Anonymity**
Data Anonymization

- From clients
Data Anonymization Questions

- What to anonymize?

Patient Records (private)

<table>
<thead>
<tr>
<th>Sex</th>
<th>Zipcode</th>
<th>DOB</th>
<th>Disease</th>
</tr>
</thead>
<tbody>
<tr>
<td>Male</td>
<td>95110</td>
<td>6/7/72</td>
<td>Heart Disease</td>
</tr>
<tr>
<td>Female</td>
<td>95110</td>
<td>1/31/80</td>
<td>Hepatitis</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Sex
Zipcode
DOB
Disease
Data Anonymization Questions

- What to anonymize?
- How to anonymize?

Patient Records (private)

<table>
<thead>
<tr>
<th>Sex</th>
<th>Zipcode</th>
<th>DOB</th>
<th>Disease</th>
</tr>
</thead>
<tbody>
<tr>
<td>Male</td>
<td>95110</td>
<td>6/7/72</td>
<td>Heart Disease</td>
</tr>
<tr>
<td>Female</td>
<td>95110</td>
<td>1/31/80</td>
<td>Hepatitis</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Sex</th>
<th>Zipcode</th>
<th>DOB</th>
<th>Disease</th>
</tr>
</thead>
<tbody>
<tr>
<td>“Unknown”</td>
<td>95***, 1972</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Masking  USA CA, USA
Generic San Jose
Random
Data Anonymization Questions

- What to anonymize?
- How to anonymize?
- How useful is the anonymized data for testing and debugging?

Patient Records (private)

<table>
<thead>
<tr>
<th>Sex</th>
<th>Zipcode</th>
<th>DOB</th>
<th>Disease</th>
</tr>
</thead>
<tbody>
<tr>
<td>Male</td>
<td>95110</td>
<td>6/7/72</td>
<td>Heart Disease</td>
</tr>
<tr>
<td>Female</td>
<td>95110</td>
<td>1/31/80</td>
<td>Hepatitis</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Sex     | Zipcode  | DOB   | Disease  |
---      | -------- | ----- | -------- |
"Unknown" | 95***, 1972 | ... | ...     |
Masking | USA     | CA, USA | San Jose |
Generic  | Generic | USA   | CA, USA  |
Random   | Random  | San Jose | USA      |
Our Solution

- \textit{kb-Anonymity}: A model that provides guidance on the anonymization questions
  
  How to anonymize
  
  - Follow guidance provided by the \textit{k-anonymity} privacy model
    - Each tuple has at least k-1 indistinguishable peers
  
  - Generate concrete values always
  
  - Remove indistinguishable tuples

  How useful is the anonymized data
  
  - Preserve utility for testing and debugging
  
  - Each anonymized tuple exhibits certain kinds of \textit{behavior} exhibited by original tuples
**kb-Anonymity**

- Behavior preservation
**kb-Anonymity**

- Privacy preservation

![Diagram showing different levels of privacy preservation from Random to No Field Repeat to No Tuple Repeat to Standard k-Anonymity to None, with a spectrum from Low to High behavior preservation.](image-url)
### kb-Anonymity

- Behavior and Privacy preservation

![Diagram showing the relationship between privacy preservation and behavior preservation for different methods.](image)

<table>
<thead>
<tr>
<th>Privacy Preservation</th>
<th>Low</th>
<th></th>
<th></th>
<th>High</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td>Same Path</td>
<td>Same Path with Input Restrictions</td>
<td>Same Program States</td>
</tr>
<tr>
<td>Standard k-Anonymity</td>
<td>N/I</td>
<td>N/I</td>
<td>N/I</td>
<td>N/I</td>
</tr>
<tr>
<td>No Tuple Repeat</td>
<td>N/I</td>
<td>✓</td>
<td>✓</td>
<td>X</td>
</tr>
<tr>
<td>No Field Repeat</td>
<td>N/I</td>
<td>✓</td>
<td>✓</td>
<td>X</td>
</tr>
<tr>
<td>None</td>
<td>N/I</td>
<td>N/I</td>
<td>N/I</td>
<td>N/I</td>
</tr>
</tbody>
</table>
\textbf{kb-Anonymity - Another View}

- Anonymization function (i.e., value replacement function) $F: R \rightarrow R$

- Each original tuple is mapped by $F$ to at most one released tuple
- At least $k$ original tuples are mapped to the same released tuple

\begin{itemize}
  \item $t_1=<f_1, \ldots, f_i, \ldots, f_n>$
  \item $t_2=<f_1, \ldots, f_i, \ldots, f_n>$
  \item \ldots
  \item $t_k=<f_1, \ldots, f_i, \ldots, f_n>$
  \item $t_1^r=<f_1, \ldots, f_i^r, \ldots, f_n>$
\end{itemize}
**kb-Anonymity Implementation**

- Dynamic symbolic (a.k.a. concolic) execution with controlled constraint generation and solving.

```
Raw Dataset
\[ t_1 = \langle f_1, \ldots, f_p, \ldots, f_n \rangle \]
\[ t_2 = \langle f_1, \ldots, f_p, \ldots, f_n \rangle \]
\[ \ldots \]
\[ t_k = \langle f_1, \ldots, f_p, \ldots, f_n \rangle \]

Program Execution

Path-Grouped Dataset

Sets of Tuples of the Same Path:
\[ t_1 = \langle f_1, \ldots, f_p, \ldots, f_n \rangle \]
\[ t_k = \langle f_1, \ldots, f_p, \ldots, f_n \rangle \]

Released Dataset

Privacy & Behavior Preserving Tuples
\[ t_{1}^r = \langle f_1, \ldots, f_{i^r}, \ldots, f_n \rangle \]
```
**kb-Anonymity Implementation**

- Dynamic symbolic (a.k.a. concolic) execution with controlled constraint generation and solving
kb-Anonymity Implementation

- Dynamic symbolic (a.k.a. concolic) execution with controlled constraint generation and solving

Raw Dataset

$t_1 = \langle f_1, \ldots, f_p, \ldots, f_n \rangle$
$t_2 = \langle f_1, \ldots, f_p, \ldots, f_n \rangle$

\[ \ldots \]

$t_k = \langle f_1, \ldots, f_p, \ldots, f_n \rangle$

Program Execution

Path-Grouped Dataset

Sets of Tuples of the Same Path:

$t_1 = \langle f_1, \ldots, f_p, \ldots, f_n \rangle$
$t_k = \langle f_1, \ldots, f_p, \ldots, f_n \rangle$

Constraint Generation (JPF+JFuzz)

Anonymized Dataset

Sets of Indistinguishable Tuples:

$t_1^a = \langle f_1, \ldots, ^*, \ldots, f_n \rangle$
$t_k^a = \langle f_1, \ldots, ^*, \ldots, f_n \rangle$

Input Restrictions

Released Dataset

Privacy & Behavior Preserving Tuples

$t_1^r = \langle f_1, \ldots, f_l, \ldots, f_n \rangle$
**kb-Anonymity Implementation**

- Dynamic symbolic (a.k.a. concolic) execution with controlled constraint generation and solving

1. **Raw Dataset**
   - $t_1 = <f_1, \ldots, f_p, \ldots f_n>$
   - $t_2 = <f_1, \ldots, f_p, \ldots f_n>$
   - $\ldots$
   - $t_k = <f_1, \ldots, f_p, \ldots f_n>$

2. **Program Execution**

3. **Path-Grouped Dataset**
   - Sets of Tuples of the Same Path:
     - $t_1 = <f_1, \ldots, f_p, \ldots f_n>$
     - $t_k = <f_1, \ldots, f_p, \ldots f_n>$

4. **Anonymized Dataset**
   - Sets of Indistinguishable Tuples:
     - $t_1^a = <f_1, \ldots, f_p, \ldots f_n>$
     - $t_k^a = <f_1, \ldots, f_p, \ldots f_n>$

5. **Constraint Generation (JPF+JFuzz)**

6. **Constraint Solver (Choco)**

7. **Released Dataset**
   - Privacy & Behavior Preserving Tuples
     - $t_1^r = <f_1, \ldots, f_p, \ldots f_n>$

8. **Input Restrictions**

9. **Anonymization Per Path Group**
Empirical Evaluation

- On slices of open source programs
  - OpenHospital, iTrust, PDManager
  - From sourceforge
  - Modified to deal with integers only
  - Randomly generated test data for anonymization
Empirical Evaluation - Utility

- 16 fields: first name, last name, age, gender, address, city, number of siblings, telephone number, birth date, blood type, mother’s name, mother’s deceased status, father’s name, father’s deceased status, insurance status, and whether parents live together.

<table>
<thead>
<tr>
<th>No</th>
<th>Raw Data Point</th>
<th>Released Tuple</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>⟨90207, 10125, 2, -1, 16261, 22549, 69883, 914, 8201, -2, 68353, -1, -53, -1, -1, -2⟩</td>
<td>⟨-9999, 10000, 0, -10000, 16261, 22549, 69883, 914, 8201, -2, 68353, -1, -53, -1, -1, -2⟩</td>
</tr>
<tr>
<td>2</td>
<td>⟨19892, 16536, 78, 1, 36688, 88797, 172, 7519, 50896, -1, 44500, 1, 7452, -2, -1, 1⟩</td>
<td>⟨-9999, 10000, 0, -10000, 16261, 22549, 69883, 914, 8201, -2, 68353, -1, -53, -1, -1, -2⟩</td>
</tr>
<tr>
<td>3</td>
<td>⟨35778, 21908, 89, -1, 89965, 41493, 35861, 50182, 79181, 1, 30668, -1, 34926, -2, -1, 1⟩</td>
<td>⟨-9999, 10000, 0, -10000, 16261, 22549, 69883, 914, 8201, -2, 68353, -1, -53, -1, -1, -2⟩</td>
</tr>
<tr>
<td>4</td>
<td>⟨9543, 23693, 48, 1, 18133, 75043, -173, 38100, 14912, 1, 69504, 0, 14969, -1, -2, 1⟩</td>
<td>⟨-9999, 10000, 0, -10000, 16261, 22549, 69883, 914, 8201, -2, 68353, -1, -53, -1, -1, -2⟩</td>
</tr>
<tr>
<td>5</td>
<td>⟨42164, 40607, -6, 1, 46920, 21328, 15089, 42147, 81975, 1, 24382, -2, -252, -2, -1, -1⟩</td>
<td>Error Message</td>
</tr>
</tbody>
</table>
Empirical Evaluation - Scalability

- Running time is proportional to the size of the original data set, and almost constant per tuple.

![Scalability Diagram](image)

- x-axis: different configurations; y-axis: running time in seconds;
- Different colors represent the sizes of different original data sets.
Limitations

- Selection of quasi-identifiers
  - Reply on data owners to choose appropriate QIs
- Assume each tuple is used independently from other tuples by a program
- Data distortion
  - Do not maintain data statistics, and thus not suitable for data mining or epidemiological studies
- Integer constraints only
  - May handle string constraints based on JPF+jFuzz
Future Work

- Model Refinement
  - Various definitions of behavior preservation
  - Various privacy models

![Diagram showing various privacy preservation models and their coverage]

Input & output

Behavior Preservation

Where is the best place to stay?

Privacy Preservation

- l-diversity
- m-invariant
- t-closeness
Related Work

On concolic execution

Related Work

On privacy-preserving testing & debugging

- James Clause and Alessandro Orso. **Camouflage: Automated Anonymization of Field Data.** In ICSE 2011.
- Mark Grechanik, Christoph Csallner, Chen Fu, and Qing Xie. **Is Data Privacy Always Good For Software Testing?** In ISSRE 2010.
Related Work

- On privacy-preserving testing & debugging

[ISSRE 2010] consider same statement coverage; focus on choosing better QIs, then use standard $k$-anonymity algorithm

[USENIX Security 2008, ASPLOS 2008, ICSE 2011] consider path conditions; focus on anonymizing a single tuple

These studies complement ours in cases when only a limited number of failed test inputs are considered.

[USENIX Security 2003] focus on anonymizing a single tuple only
Conclusion

- *kb*-Anonymity: A model that guides data anonymization for software testing and debugging purposes.
Thank you!

Questions?
{adityabudi, davidlo, lxjiang, lucia.2009}@smu.edu.sg