

# Privacy-Enhancing $k$ -Anonymization of Customer Data

Sheng Zhong   Zhiqiang Yang   **Rebecca Wright**  
Stevens Institute of Technology

Bertinoro Privacy Workshop  
13 July, 2005

*Appears in 24th ACM Symposium on Principles of Databases.*



# Anonymization to Protect Privacy?

Date of Birth	Zip Code	Allergy	History of Illness
03-24-79	07030	Penicillin	Pharyngitis
08-02-57	07028	No Allergy	Stroke
11-12-39	07030	No Allergy	Polio
08-02-57	07029	Sulfur	Diphtheria
08-01-40	07030	No Allergy	Colitis



Medical Research  
Database

Sensitive  
Information

# Risk of Re-identification

## Quasi-identifiers

Date of Birth	Zip Code	Allergy	History of Illness
08-02-57	07028	No Allergy	Stroke
11-12-39	07030	No Allergy	Polio
08-02-57	07029	Sulfur	Diphtheria
08-01-40	07030	No Allergy	Colitis



*I know Victor is in this table, and I know his birthday is 08-02-57 and he lives in the 07028... Now I've learned he has a history of stroke!*

# *k*-Anonymity [SS98]

**Idea:** Make re-identification more difficult, by ensuring that there are at least  $k$  records with a given quasi-id.

Date of Birth	Zip Code	Allergy	History of Illness
*	07030	Penicillin	Pharyngitis
08-02-57	0702*	No Allergy	Stroke
*	07030	No Allergy	Polio
08-02-57	0702*	Sulfur	Diphtheria
*	07030	No Allergy	Colitis

**2-anonymous table**

# Property of $k$ -Anonymous Table

- Each value of quasi-identifier attributes appears  $\geq k$  times in the table (or it does not appear at all)
- Each row of the table is hidden in  $\geq k$  rows
- Each person involved is hidden in  $\geq k$  peers

# *k*-Anonymity May Protect Privacy

	Date of Birth	Zip Code	Allergy	History of Illness
08-02-57	0702*	No Allergy	Stroke	
	08-02-57	0702*	No Allergy	Stroke
08-02-57	0702*	Sulfur	Diphtheria	
	*	07030	No Allergy	Colitis



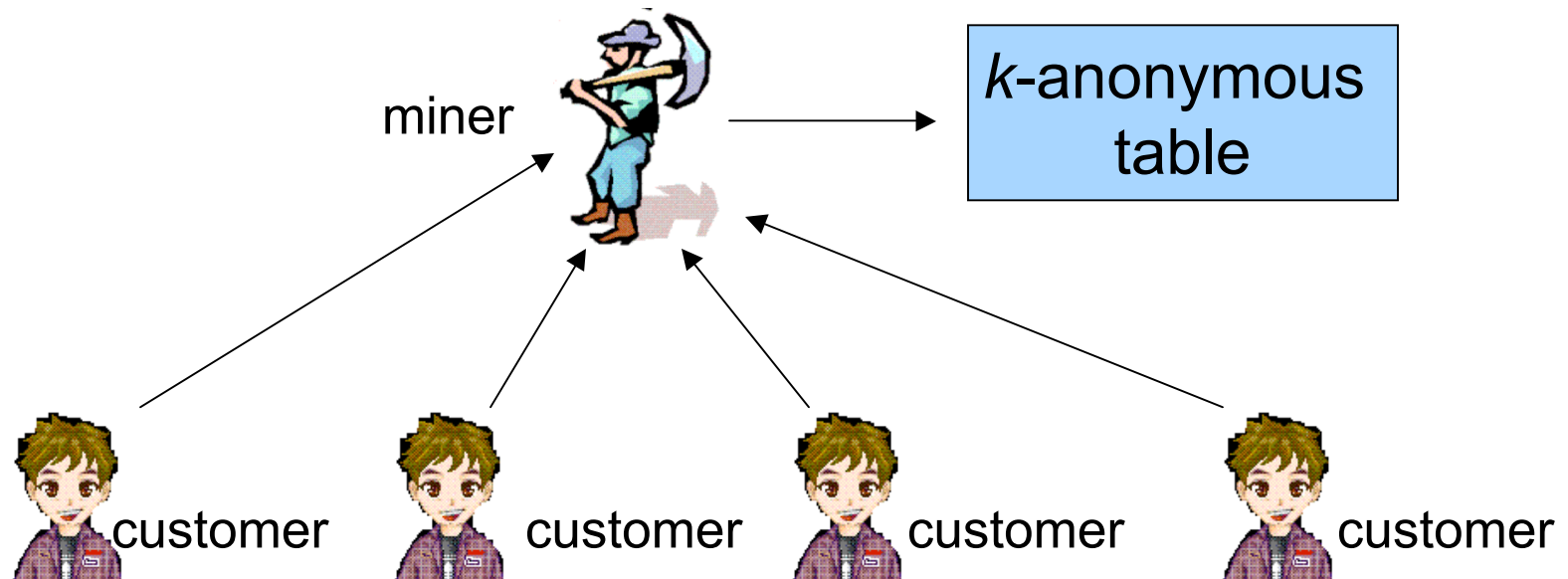
*Which of them is Victor's record?  
Confusing...*

# Centralized $k$ -Anonymization

- $k$ -anonymity has been extensively studied [Sam01, Swe02a, Swe02b, MW04, AFK+04, BA05].
- Existing  $k$ -anonymization algorithms assume a single party that has access to the entire original table.
- We do not make this assumption, thereby providing additional privacy.

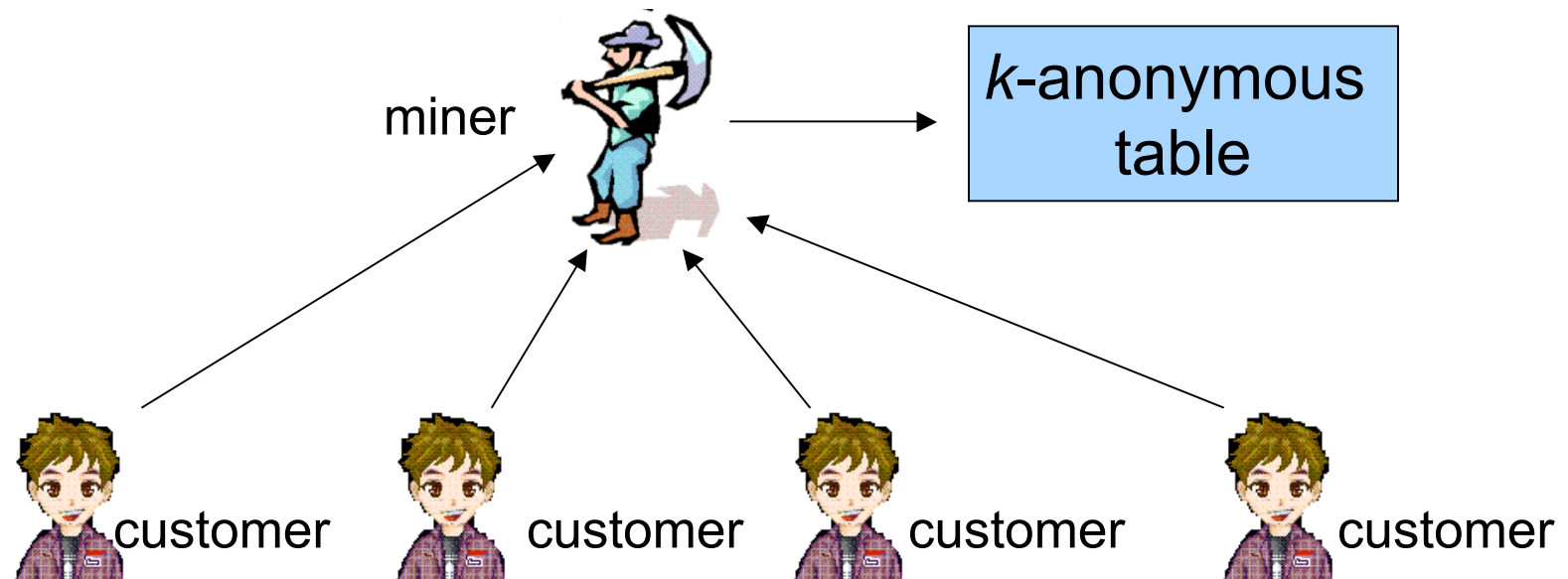
# Distributed $k$ -Anonymization

- $N$  customers + 1 publisher (or miner)
- Each user/respondent/customer has her personal data, comprising a row of the table.



# Distributed $k$ -Anonymization

**Privacy Goal:** The miner should not be able to associate sensitive information in the table with the corresponding customer.



# Formulations of Problem

- We can formulate the problem differently by considering different methods to  $k$ -anonymize the table.
- We consider two formulations:

<b>Problem Formulation</b>	<b><math>k</math>-Anonymization Method</b>	<b>Privacy Protection</b>
Formulation 1	Extracting the $k$ -anonymous part	Hiding sensitive information outside this part
Formulation 2	MW algorithm [MW04]	Hiding quasi-identifiers suppressed by MW algorithm

# Problem Formulation 1

*k*-anonymous

part: largest subset of rows that is *k*-anonymous

Date of Birth	Zip Code	Allergy	History of Illness
11-12-39	07030	No Allergy	Colitis
08-02-57	07029	No Allergy	Stroke
03-24-79	07030	Penicillin	Pharyngitis
08-02-57	07029	Sulfur	Diphtheria
11-12-39	07030	No Allergy	Colitis

**Privacy Guarantee:** The miner should not learn the privacy-sensitive attributes of the rows outside the *k*-anonymous part.

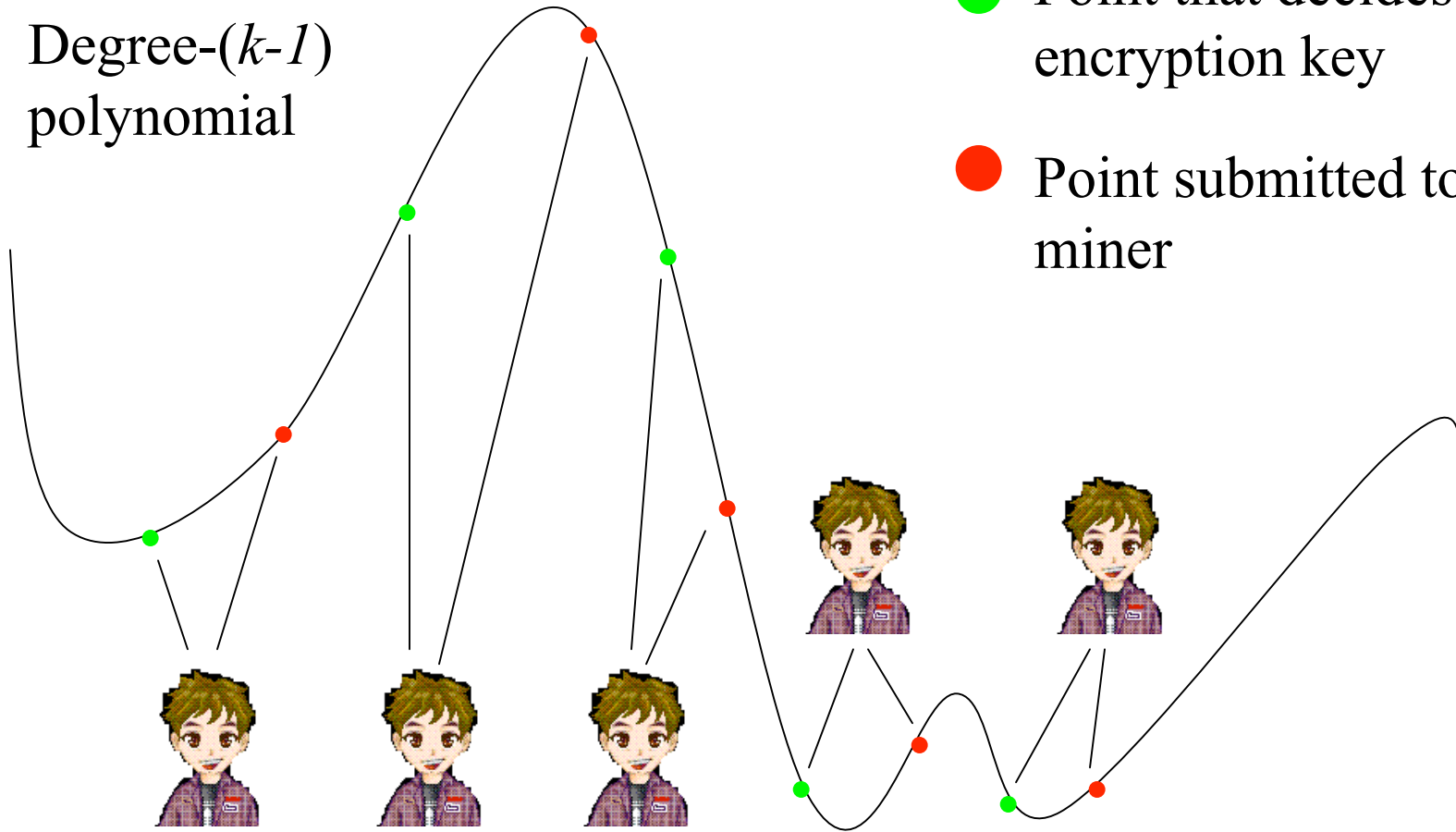
# Basic Idea of Solution

- Each customer encrypts her sensitive attributes using an encryption key that can be derived by the miner if and only if there are  $\geq k$  rows whose quasi-identifiers are equal. No customer knows others' encryption key.
- $\Rightarrow$  the miner is able to see the sensitive attributes if and only if there are  $\geq k$  customers whose quasi-identifiers are equal.
- Uses  $(2N, k)$ -Shamir secret sharing, hash functions, and ElGamal-like threshold encryption with key reconstruction via La Grange interpolation in the exponent.

# Illustration of Solution

Degree- $(k-1)$   
polynomial

- Point that decides encryption key
- Point submitted to miner



# Summary of Formulation 1

- Miner learns only those rows of the table for which the same quasi-identifier appears at least  $k$  times.
- Therefore, this solution is mainly applicable when the data is already close to  $k$ -anonymous.
- Customers may agree to generalize data before submission in order to increase likelihood of  $k$ -anonymity.
  - For example, use birth year not entire birth date.

# Formulation 2

- In this formulation, the miner learns a table that suppresses some quasi-ids of the original table in order to  $k$ -anonymize it.
- Our solution is a distributed privacy-preserving version of the [MW04] algorithm.
- It protects miner from learning the values of the suppressed quasi-ids. *Miner does learn certain additional information.*

# Problem Formulation 2

Date of Birth	Zip Code	Allergy	History of Illness
03-24-79	07030	Penicillin	Pharyngitis
08-02-57	07030	No Allergy	Stroke
11-12-39	07030	No Allergy	Polio
08-02-57	07030	Sulfur	Diphtheria
08-01-40	07030	No Allergy	Colitis

MW algorithm



**Privacy Guarantee:** The miner should not learn the quasi-identifiers suppressed by MW algorithm.

Date of Birth	Zip Code	Allergy	History of Illness
*	07030	Penicillin	Pharyngitis
08-02-57	07030	No Allergy	Stroke
*	07030	No Allergy	Polio
08-02-57	07030	Sulfur	Diphtheria
*	07030	No Allergy	Colitis

# MW Algorithm

- **Phase 1:** Compute the distance between each pair of rows.
  - The distance between two rows is the number of quasi-id attributes in which they have different entries
- **Phase 2:** Compute a  $k$ -partition of the table.
  - A  $k$ -partition is a collection of disjoint subsets of rows in which each subset contains at least  $k$  rows and the union of these subsets is the entire table.
- **Phase 3:** Compute the  $k$ -anonymized table.
  - Replace any differing entries in each  $k$ -partition with \*'s.

# Private Distributed Algorithm

- **Phase 1:** Compute the distance between each pair of rows.
  - We give a protocol for the miner to learn these distances without learning the quasi-ids.
- **Phase 2:** Compute a  $k$ -partition of the table.
  - Miner can do this locally using distances from Phase 1.
- **Phase 3:** Compute the  $k$ -anonymized table.
  - We give a protocol by which the miner learns the quasi-ids that do not need to be suppressed.

# Phase 1:

Computing distances between rows

- For row  $i$  and row  $i'$  and the  $j$ th quasi-id attribute, we define:

$$same(i, i', j) = \begin{cases} 1 & \text{if the two rows have the same} \\ & \text{value in this quasi-id attribute} \\ random & \text{otherwise} \end{cases}$$

⇒ With high probability, the distance between row  $i$  and row  $i'$  is the number of 1's in

$$\{same(i, i', j) : j \in [1, m]\}$$

# Computing Distances

- In a prime-order cyclic group, we have

$$\mathit{same}(i, i', j) = \left( \frac{j\text{th quasi-id of } i\text{th row}}{j\text{th quasi-id of } i'\text{th row}} \right)^{\mathit{random}}$$

⇒ Only need to compute the quotient of quasi-ids.

# Privately Computing Quotient

- Use a multiplicatively homomorphic encryption scheme.
  - Encryption of quotient can be easily derived from encryptions of quasi-ids.
- Customers jointly help the miner to decrypt the quotient.
  - Existing technique to achieve this: threshold decryption [DF89].
- This completes Phase 1.

# Phase 2

- In this phase, the miner follows the MW algorithm, which is a heuristic algorithm for learning a good  $k$ -partition from the pairs of inter-row distances.
- No interaction with the customers is needed in this phase.

# Phase 3:

Computing  $k$ -anonymized table from  $k$ -partition

- Output of Phase 2 is a partition of the table into subsets of rows.
- In each subset  $S$  and for each quasi-id attribute:
  - if all rows in  $S$  agree on this quasi-id, miner needs only to learn this quasi-id value.
  - otherwise, miner should not learn the quasi-id values, which should be replaced by \*.

# Comparing Quasi-id in $\mathcal{S}$

- With all but negligible probability, all rows in a subset  $S$  agree on  $j$ th quasi-id if and only if

$$\prod_{i, i' \in S} \text{same}(i, i', j) = 1$$

- Miner can compute an encryption of the above product using the homomorphic property.
- Customers help with threshold decryption in a way that miner learns either the quasi-id (if there are at least  $k$ ) or \* (otherwise).

# Summary of Formulation 2

- This solution is a distributed privacy-preserving version of the MW algorithm.
- It works well even if the original table is not close to  $k$ -anonymous.
- However, the customers must be willing to allow the miner to learn the inter-row distances.

# Privacy of $k$ -anonymity

- It is clear that if  $k_1 > k_2$ , then  $k_1$ -anonymous data is not less private than  $k_2$ -anonymous data. (Well, maybe. Depends on exact partitions.)
- Can we make any stronger statements? Difficulties:
  - An attacker may have additional information that helps reidentify exactly or approximately.
  - Decision about which attributes constitute quasi-ids may have been wrong.

# Conclusion

- It is often desirable to make data public for various purposes.
- But, there are privacy concerns about this data.
- $k$ -anonymity has been proposed to provide a balance between privacy and utility.
- In our work, we enhance the end-to-end privacy of  $k$ -anonymity by eliminating the need for data to be collected in a single place.