Distributed Privacy Preserving Data Collection

Mingqiang Xue[1], Panagiotis Papadimitriou[2], Chedy Raïssi[3], Panos Kalnis[4], and Hung Keng Pung[1]

[1] Computer Science Department, National University of Singapore
[2] Stanford University
[3] INRIA Nancy
[4] King Abdullah University of Science and Technology

Abstract. We study the distributed privacy preserving data collection problem: an untrusted data collector (e.g., a medical research institute) wishes to collect data (e.g., medical records) from a group of respondents (e.g., patients). Each respondent owns a multi-attributed record which contains both non-sensitive (e.g., quasi-identifiers) and sensitive information (e.g., a particular disease), and submits it to the data collector. Assuming T is the table formed by all the respondent data records, we say that the data collection process is privacy preserving if it allows the data collector to obtain a k-anonymized or l-diversified version of T without revealing the original records to the adversary.

We propose a distributed data collection protocol that outputs an anonymized table by generalization of quasi-identifier attributes. The protocol employs cryptographic techniques such as homomorphic encryption, private information retrieval and secure multiparty computation to ensure the privacy goal in the process of data collection. Meanwhile, the protocol is designed to leak limited but non-critical information to achieve practicability and efficiency. Experiments show that the utility of the anonymized table derived by our protocol is in par with the utility achieved by traditional anonymization techniques .

1 Introduction

In the data collection problem a third party collects data from a set of individuals who concern about their privacy. Specifically, we consider a setting in which there is a set of data *respondents*, each of whom has a row of a table, and a data *collector*, who wants to collect all the rows of the table. For example, a medical researcher may request from some patients that each of them provides him with a health record that consists of three attributes: $\langle$*age*, *weight*, *disease*$\rangle$. Figure 1(a) shows the table of the patients' records.

Although the health record contains no explicit identifiers such as name and phone numbers, an adversarial medical researcher may be able to retrieve a patient's identity using the combination of *age* and *weight* with external information. For instance, in the data records of Figure 1(a), we see that there is only one patient with age 45 and weight 60 and this patient suffers from Gastritis (the third row). If the researcher knows a particular patient with the same age and weight values, after collecting all the data records he learns that this patient suffers from Gastritis. In this case the attributes *age* and *weight* serve as a quasi-identifier. Generally, the patients feel comfortable to provide the researcher with medical records only if there is a guarantee that the researcher can

J.X. Yu, M.H. Kim, and R. Unland (Eds.): DASFAA 2011, Part I, LNCS 6587, pp. 93–107, 2011.

only form an anonymized table with their records. In k-anonymity [16], each record has at least $k-1$ other records whose values are indistinct over the quasi-identifier attributes [16]. l-diversity [10] further requires that there are at least l well represented sensitive values for records in the same *equivalence class*. The patients may achieve k-anonymity or l-diversity by generalizing the values that correspond to the quasi-identifiers [14]. In Figure 1(b), observe that if each patient discloses only some appropriate range of his age and weight instead of the actual values, then the medical researcher sees a 4-anonymous and 3-diverse table. In this case, the researcher can only determine with probability at most 1/2 the disease of the 45-year old patient.

In the privacy preserving data collection the data respondents look for the minimum possible generalization of the quasi-identifier values so that the collector receives an anonymized table. The constraint of the problem is that although the respondents can communicate with each other and with the collector, no single participant can leak any information to the others except from his final anonymous record. Traditional table anonymization techniques [16] are not applicable to our problem, as they assume that there is a single trusted party that has access to all the table records. If the trusted party is compromised then the privacy of all respondents is compromised as well. In our approach, each respondent owns his own record and does not convey its information to any other party prior to its anonymization.

Our setting is similar to the distributed data collection scenario studied by Zhong et al [19]. The difference is that in their work the respondents create a k-anonymous table for the collector by suppressing quasi-identifier attribute values. We use generalization instead of suppression, which makes the problem not only more general but also much more practical. Our problem is more general because suppression is considered as a special case of generalization: a suppressed attribute value is equivalent to the value generalization to the higher level of abstraction. The problem is also more practical because generalized attribute values have greater utility than suppressed values, since they convey more information to the data collector without compromising the respondents' privacy. Moreover, our solution not only achieves k-anonymity, but also l-diversity. Our contributions are the following:

- We formally define the problem of distributed privacy preserving data collection with respondents that can generalize attribute values.
- We present an efficient and privacy-preserving protocol for k-anonymous or l-diverse data collection.
- We show theoretically the information leakage that our protocol yields.
- We evaluate our protocol experimentally to show that it achieves similar utility preservation as the state-of-the art non-distributed anonymization algorithm [6].

2 Related Work

In [19], the authors proposed a distributed, privacy-preserving version of the *MW* [11], which is an $O(k \log k)$ approximation to optimal k-anonymity based on entry suppression; in contrast, our algorithm supports generalization. Similar to our scheme, in order to achieve efficient distributed anonymization the distributed MW algorithm reveals information about the relative distance between different data record pairs. In [19], the

Age	Weight	Disease
35	50	Gastritis
40	55	Diabetes
45	60	Gastritis
45	65	Pneumonia
55	65	Gastritis
60	60	Diabetes
60	55	Diabetes
65	50	Alzheimer
55	75	Diabetes
60	75	Flu
65	85	Flu
70	80	Alzheimer

(a) Original

Age	Weight	Disease
[35, 45]	[50, 65]	Gastritis
[35, 45]	[50, 65]	Diabetes
[35, 45]	[50, 65]	Gastritis
[35, 45]	[50, 65]	Pneumonia
[55, 65]	[50, 65]	Gastritis
[55, 65]	[50, 65]	Diabetes
[55, 65]	[50, 65]	Diabetes
[55, 65]	[50, 65]	Alzheimer
[55, 70]	[75, 85]	Diabetes
[55, 70]	[75, 85]	Flu
[55, 70]	[75, 85]	Flu
[55, 70]	[75, 85]	Alzheimer

(b) Anonymized

Fig. 1. Distributed medical records table

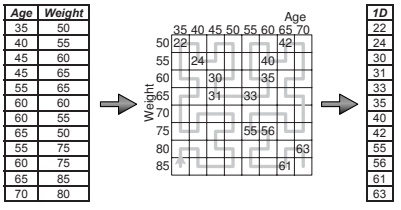


Fig. 2. Mapping 2D to 1D points using Hilbert curve

distance between two records is the *number* of differences in the attribute values. For example, in Figure 1(a), the distance between the first two records is 2, since age 35 is different from age 40 and weight 50 is different from weight 55. In our approach, the distance between two records depends on the *distance* between the corresponding attribute values, which is more difficult to evaluate securely. In [18] the authors proposed another k-anonymous data collection scheme. Opposed to the work in [19], this scheme has eliminated the need for unidentified communication channel by the data respondent. However, this scheme is still based on entry suppression and it is unclear whether the scheme can be generated to l-diversity. A similar approach appears in [7]. They considered distributed data collection problem based on a suppression based k-anonymity algorithm Mondrian and they only consider the k-anonymity case. Different from the above, [3] and [17] considered the anonymity-preserving data collection problem. Although, the setting of the problem is similar to ours, their protocols aims to allow the miner to collect original data from the respondents without linking the data to individuals. When the data collected contains identifiable information as in our case, their solutions are not applicable.

The anonymization algorithm that we present in this paper is based on the the Fast data Anonymization with Low information Loss (*FALL*)*et al* [6]. In this work, efficient anonymization is achieved in two steps. The first step includes the transformation of u-dimensional to 1-dimensional data, in which a multi-attributed data record is converted to an integer using a space filling curve (e.g. Hilbert curve [12]). For example, Figure 2 shows a Hilbert walk that visits each cell in the two dimensional space ($Weight \times Age$) and assigns each cell with an integer in increasing order along the walk. In the second step, an optimal 1D k-anonymization is performed over the set of integers obtained in the first step using an efficient algorithm based on dynamic programming. The same partitions will be used for forming the *equivalence classes* of data records. Similarly, efficient l-diversity can be achieved using heuristics in a similar manner as k-anonymity.

3 Problem Formulation

3.1 The System and the Adversaries

The system employs the Client-Server architecture. Each respondent runs a client. There is an untrusted server that facilitates the communication and computation in the system

on behalf of the collector. We assume that all messages are encrypted, and secure communication channels exist between any pair of communicating parties. By the end of the protocol execution, an anonymized table, generalized from the data records of the respondents, is created at the server side (i.e., the collector).

The adversaries can either be the respondents or the server. We assume that the adversaries follow the semi-honest model, which means that they always correctly follow the protocol but are curious in gaining additional information during the execution of the protocol. In addition, we assume that the adversarial respondents can collaborate with each other to gain additional information. We assume there can be up to $t_{ss} - 1$ adversaries among the respondents, where t_{ss} is a security parameter.

3.2 Notion of Privacy

Initially, there are x number of respondents each running an instance of the client. We denote the set of non-sensitive attributes of the data records $A = \{a_1, a_2, \ldots, a_u\}$ and the sensitive attribute s^i. The data record for the i^{th} respondent is represented as $t_i = \{a_1^i, \ldots, a_u^i, s^i\}$ and $T = \{t_1, t_2, \ldots, t_x\}$ is the table formed by the original data records of the respondents. $t_i.A$ represents the non-sensitive attribute values for the data record t_i. Similarly, $T.A$ represents the non-sensitive attribute columns of table T. Let $\mathcal{K}(T)$ denote the final output of the protocol, which is an anonymized table generalized from T. Let $\mathcal{L}_i$ and $\mathcal{L}_{svr}$ denote the amount of information leaked in the process of protocol execution to the respondent i and the server, respectively.

During the execution of the protocol, the view of a party uniquely consists of four objects: (i) the data owned by the party, (ii) the assigned key shares, (iii) the set of received messages and (iv) all the random coin flips picked by this party. Let $\mathsf{view}_i(T)$ (respectively $\mathsf{view}_{svr}(T)$) denote the view of the respondent i (respectively the view of server). We adopt a similar privacy notion as in [19]:

Definition 1. *A protocol for k-anonymous data collection leaks only $\mathcal{L}_i$ for the respondent i and $\mathcal{L}_{svr}$ for the server if there exist probabilistic polynomial-time simulators M_{svr} and $M_1, M_2, \ldots, M_x$ such that:*

$$\{M_{svr}(\mathit{keys}_{svr}, \mathcal{K}(T), \mathcal{L}_{svr})\}_T \equiv_c \{\mathit{view}_{svr}(T)\}_T \tag{1}$$

and for each $i \in [1, x]$,

$$\{M_i(\mathit{keys}_i, \mathcal{K}(T), \mathcal{L}_i)\}_T \equiv_c \{\mathit{view}_i(T)\}_T \tag{2}$$

The contents of $\mathcal{L}_{svr}$ and $\mathcal{L}_i$ are statistical information about the respondent's quasi-identifiers. Later in this paper, we prove that the execution of our proposed protocol respects the previous definition by only leaking $\mathcal{L}_{svr}$ and $\mathcal{L}_i$ for each i.

3.3 Using Secret Sharing

To conquer up to $t_{ss} - 1$ collaborating adversaries among the respondents, we initially assume that there is a global private key SK shared by all the respondents and the server using a $(t_{ss}, x + 1)$ threshold secret sharing scheme [15]. The shares owned by

the respondents and the server are denoted as sk_1, sk_2, ..., sk_x, and sk_{svr}, respectively. With a $(t_{ss}, x+1)$ secret sharing scheme, t_{ss} or more key shares are necessary in order to successfully reconstruct the decryption function with the secret key SK, while less than t_{ss} key shares give absolutely no information about SK. The corresponding public key of the private key SK is denoted as PK. The public key encryption algorithm that we use in this paper is the Paillier's cryptosystem [13] because of its useful additive homomorphic property. To support threshold secret sharing, we use a threshold version of Paillier's encryption as described in [8] based on Asmuth-Bloom secret sharing [1].

4 Towards the Solution

4.1 A Sketch of the Solution

Preparation stage. The main goal of this stage is to map the uD records to 1D integers. In this stage, each respondent independently performs uD to 1D mapping using a space filling curve, e.g., the Hilbert curve. Symbolically, the mapping for $t_i.A$ is denoted as $c_i = \mathcal{S}(t_i.A)$. Each integer c_i is in the range $[1, c_{max}]$, where c_{max} denotes the maximum possible value that the mapping function can yield. The set of mapped values is denoted as $S = \{c_1, c_2, \ldots, c_x\}$. Without loss of generality, we assume that the values in S are already sorted in ascending order for the ease of subsequent discussion.

Stage 1. The goal of this stage is to achieve *p-probabilistic locality preserving* mapping. Symbolically, the i^{th} respondent maps the secret integer c_i to a real number $r^+_{c_i}$ using function $\mathcal{F}()$, i.e. $r^+_{c_i} = \mathcal{F}(c_i)$. Note that, although the encryption algorithm that we use do not support encryption of real numbers directly, we can use integers in the chosen field to simulate a fixed point real number which is sufficient for our purpose. The set of mapped values for all the respondents is represented as $\mathcal{F}(S) = \{r^+_{c_1}, r^+_{c_2}, \ldots, r^+_{c_x}\}$. We require that the mapping from each c_i to $r^+_{c_i}$ by $\mathcal{F}()$ preserves certain order and distance relations for the integers in S for utility efficient anonymization, which is known as *p-probabilistic locality preserving* and is defined as follows:

Definition 2. *Given any two pre-images c_{i_1}, c_{i_2}, a mapping function $\mathcal{F}()$ is* order preserving *if:*

$$c_{i_1} \leq c_{i_2} \Rightarrow \mathcal{F}(c_{i_1}) \leq \mathcal{F}(c_{i_2}) \tag{3}$$

Given any three pre-images $c_{i_1}, c_{i_2}, c_{i_3}$, and the distances $dist_1 = |c_{i_1} - c_{i_2}|$, $dist_2 = |c_{i_2} - c_{i_3}|$, a mapping function $\mathcal{F}()$ is p-probabilistic distance preserving *if:*

$$dist_1 \leq dist_2 \Rightarrow \Pr(fdist_1 \leq fdist_2) \geq p \tag{4}$$

and it increases with $dist_2$, where $fdist_1 = |\mathcal{F}(c_{i_1}) - \mathcal{F}(c_{i_2})|$, $fdist_2 = |\mathcal{F}(c_{i_2}) - \mathcal{F}(c_{i_3})|$, and p is a parameter in the $[0, 1]$.

A mapping function $\mathcal{F}()$ is *p-probabilistic locality preserving* if it is both *order preserving* and *p-probabilistic distance preserving*. In addition, we also require that the mapping from c_i to $r^+_{c_i}$ reveals limited information about c_i, which is to be γ-concealing:

Definition 3. *Given the pre-image c_i and $r^+_{c_i} = \mathcal{F}(c_i)$, the function $\mathcal{F}()$ is γ-concealing if $\Pr(c_{mle} = c_i | r^+_{c_i}) \leq 1 - \gamma$ for the Maximum Likelihood Estimation (*MLE*) c_{mle} of c_i.*

Stage 2. The goal of this stage is to determine a set of partitions of respondents based on the set of values in $\mathcal{F}(S)$ using 1D optimal k-anonymization algorithm or the l-diversity heuristics as proposed in *FALL*.

Stage 3. The goal of this stage is to privately anonymize the respondent data records based on the partitions from *Stage 2*, which involves secure computation of *equivalence classes* for the respondents in the same partition. As $\mathcal{F}(S)$ is *p-probabilistic locality preserving*, if we use the same partitions created on $\mathcal{F}(S)$ to anonymize T, we expect that the anonymized table $\mathcal{K}(T)$ preserves the utility well.

4.2 Technical Details

Stage 1. Probabilistic Locality Preserving Mapping. The challenge of performing *p-probabilistic locality preserving* mapping in this application is that all the data values in S are distributed, and we must ensure the secrecy of c_i for respondent i in the mapping process. In our approach we build an encrypted index $E(R+) = \{E(r^+_1), \ldots, E(r^+_{c_{max}})\}$ on the server side containing c_{max} randomly generated numbers that correspond to all integers in the range $[1, c_{max}]$ of the mapping function $\mathcal{S}$. Later, each respondent i retrieves then the c^{th}_i item in the encrypted index, i.e., the item $E(r^+_{c_i})$, in a private manner and can jointly and safely decrypt it with other respondents in order to build the anonymized data.

Essentially, four steps are needed in order to achieve *p-probabilistic locality preserving mapping*: *Step 1.* Two sets of encrypted real numbers are created at the server side, i.e. $E(R_{init})$ and $E(R_p)$. *Step 2.* The set of encrypted real numbers $E(R+)$ is created in a recursive way using the two sets of encrypted real numbers from *Step 1*: the set $E(R_{init})$ is used to define the value of the first encrypted number $E(r^+_1)$ and the set $E(R_p)$ is used to define number $E(r^+_i)$ in terms of $E(r^+_{i-1})$. *Step 3.* Respondent i retrieves the c^{th}_i item from index $E(R+)$ created in *Step 2* using a *private information retrieval* scheme. *Step 4:* The retrieved encrypted item is jointly decrypted by t_{ss} parties, and uploaded to the server. Its plaintext is defined as $r^+_{c_i}$, i.e., the image of c_i under $\mathcal{F}()$. In the following, we describe the above four steps in detail. In *Step 1*, we first describe how to create one encrypted random real number whose plaintext value is not known by any parties. The creation of two sets of encrypted real numbers is just a simple repetition of this process.

In order to hide the value of a random number, each of these is jointly created by both a respondent and the server. To compute an encrypted joint random number $E(r)$, the respondent randomly selects a real number r_{dr} from a uniform distribution in the interval $[\rho_{min}, \rho_{max}]$. Then the respondent sends the encrypted number $E(r_{dr})$ to the server. The server independently chooses another random real number r_{svr} from the same interval $[\rho_{min}, \rho_{max}]$ and encrypts it to obtain $E(r_{svr})$. The join of the two encrypted real numbers is computed as $E(r) = E(r_{dr}) \cdot E(r_{svr}) = E(r_{dr} + r_{svr})$ by

the additive homomorphic property of the Paillier's encryption (assuming a large modulus N is used so that round up does not take place). We are aware that both the respondent i and the server knows the range information about r. We denote such range knowledge about the joint random numbers for respondent i and the server as $\mathcal{RG}_i$ and $\mathcal{RG}_{svr}$, respectively. Recall that $\mathcal{L}_{svr}$ and $\mathcal{L}_i$ are the information leakage for the server and the data respectively. Therefore, we have that $\mathcal{RG}_{svr} \in \mathcal{L}_{svr}$ and $\mathcal{RG}_i \in \mathcal{L}_i$.

With the above technique, the first encrypted set of joint random numbers that we create is $E(R_{init}) = \{E(\iota_1), E(\iota_2), E(\ldots), E(\iota_b)\}$, where the size b is a security parameter of the system. Each of the encrypted joint random numbers is created by the server and a randomly selected respondent. The second set of encrypted joint random numbers that we create on the server side is $E(R_p) = \{E(r_1), E(r_2), \ldots, E(r_{c_{max}})\}$. To create $E(R_p)$, each respondent needs to generate $\lfloor \frac{c_{max}}{x} \rfloor$ or $\lceil \frac{c_{max}}{x} \rceil$ encrypted joint random numbers with the server, if we distribute this task evenly.

In *Step 2*, to build an encrypted set of real numbers $E(R^+)$ = $\{E(r_1^+)$, $E(r_2^+)$, ..., $E(r_{c_{max}}^+)\}$ whose plaintexts values are in ascending order based on $E(R_{init})$ and $E(R_p)$, we once again use the additive homomorphic property of Paillier's encryption:

$$\begin{cases} E(r_i^+) = E(r_i) \cdot \prod_{j=1}^{b} E(\iota_j) & i = 1 \\ E(r_i^+) = E(r_{i-1}^+) \cdot E(r_i) & i = 2, \ldots, c_{max} \end{cases} \tag{5}$$

In *Step 3*, $E(r_{c_i}^+)$ is retrieved from the server by the respondent i who owns the secret c_i using Private Information Retrieval (*PIR*) scheme. We adopt the single database *PIR* scheme developed in [5] which supports the retrieval of a block of bits with constant communication rate. This *PIR* scheme is proven to be secure based on a simple variant of the Φ-hiding assumption. To hide the complexity of the *PIR* communications, we use the $\mathcal{PIR}(c_i, E(R^+))$ to represent the sub-protocol that privately retrieves the $c_i{}^{th}$ item in the set $E(R^+)$ by the i^{th} respondent, and the result of retrieval is $E(r_{c_i}^+)$.

In *Step 4*, after the respondent i has retrieved $E(r_{c_i}^+)$, he partially decrypts $E(r_{c_i}^+)$ and sends the partially decrypted cipher to $t_{ss} - 2$ other respondents for further decryption. The last partial decryption is done by the server, after which the server obtains the plaintext $r_{c_i}^+$. Note that the server cannot identify the value of c_i by re-encrypting the $r_{c_i}^+$ and search through $E(R^+)$, as the Paillier's encryption is a randomized algorithm in which the output ciphers are different for the same plaintext with different random inputs. Finally, we have achieved the mapping from the c_i to $r_{c_i}^+$. The server obtains the set $\mathcal{F}(S)$ by the end of this step.

We illustrate these four steps in the Figure 3. The first column describes the respondents 1D data. The second column represents the 33^{rd} to 40^{th} entries in $E(R^+)$. The third column represents 33^{rd} to 40^{th} entries in $E(R_p)$. The i^{th} entry of $E(R^+)$ is computed based on the product of the $(i-1)^{th}$ entry of $E(R^+)$ and the i^{th} entry of $E(R_p)$. For example, $E(r_{34}^+) = E(r_{33}^+) \cdot E(r_{34})$ by the additive homomorphic property, $E(r_{34}^+) = E(r_{33}^+ + r_{34})$ which translated in terms of real values gives $E(304.7) = E(293.5) \cdot E(11.2) = E(293.5 + 11.2)$.

Theorem 1. *The mapping function* $\mathcal{F}()$ *is* $\frac{1}{2}$-probabilistic locality preserving.

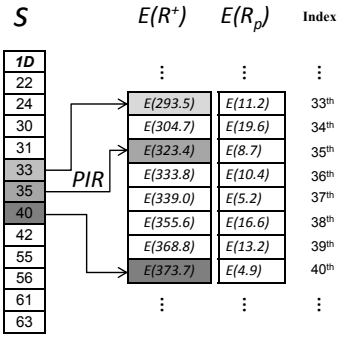


Fig. 3. Example of the probabilistic locality preserving mapping construction

Proof. Since R_p^+ is a set of ascending real numbers, we have $r^+_{c_{i_1}} \leq r^+_{c_{i_2}}$, if $c_{i_1} \leq c_{i_2}$. Therefore, $\mathcal{F}()$ is order preserving by Equation 3. To prove that it is also $\frac{1}{2}$-*probabilistic distance preserving*, let c_{i_1}, c_{i_2}, c_{i_3} be any randomly selected pre-images, and $dist_1$, $dist_2$, $fdist_1$ and $fdist_2$ follow the definitions in Definition 2 Equation 4. Assume that $c_{i_1} \leq c_{i_2} \leq c_{i_3}$ and $dist_1 \leq dist_2$. The exact form of the distributions of $fdist_1$ and $fdist_2$ are difficult to estimate. However, since $fdist_1$ ($fdist_2$ *resp.*) is the sum of $dist_1$ ($dist_2$ *resp.*) number of joint random numbers, where each joint random number is the sum of two random uniformly selected real numbers in the interval $[\rho_{min}, \rho_{max}]$, $fdist_1$ and $fdist_2$ can be unbiasedly approximated by continuous normal distribution according to the *central limit theorem*. Let $\mu = \frac{\rho_{min}+\rho_{max}}{2}$ and $\sigma^2 = \frac{(\rho_{min}-\rho_{max})^2}{12}$ be the mean and variance of the uniform distribution respectively, and without ambiguity, $fdist_1$ and $fdist_2$ be the continuous random variables. From the *central limit theorem*, we have $fdist_1 \sim N(dist_1 \cdot 2\mu, dist_1 \cdot 2\sigma^2)$ and $fdist_2 \sim N(dist_2 \cdot 2\mu, dist_2 \cdot 2\sigma^2)$. Therefore, $fdist_1 - fdist_2 \sim N((dist_1 - dist_2) \cdot 2\mu, (dist_1 + dist_2) \cdot 2\sigma^2)$. From the property of continuous normal distribution, $\Pr(fdist_1 - fdist_2 \leq 0) = \Pr(fdist_1 \leq fdist_2) \geq \frac{1}{2}$ when $dist_1 \leq dist_2$ and it increases with $dist_2$. Hence, by Equation 4, $\mathcal{F}()$ is also $\frac{1}{2}$-*probabilistic distance preserving*.

Stage 2. Anonymization in the mapped space. Suppose the anonymization algorithm in *FALL*(i.e. the 1D optimal k-anonymization or l-diversity heuristics) is used by the server for determining the partitions. Let $Z = \{z_1, z_2, \ldots, z_\pi\}$ be the result of anonymization, where the i^{th} element in Z is the ending index of the i^{th} partition of respondents and there are π number of partitions. Without losing generality, we assume the indices in Z are sorted in ascending order.

Stage 3. Secure computation of equivalence classes. In this stage, the quasi-identifiers of respondents in the same partition defined by Z form an equivalence class in $\mathcal{K}(T)$. Consider the i^{th} partition defined by Z, which is formed by the $z_{i+1} - z_i$ number of respondents with IDs $z_i, z_i + 1, \ldots, z_{i+1} - 1$, where $k \leq z_{i+1} - z_i \leq 2k - 1$. Note that each non-sensitive attribute in the partition will be generalized to an interval in the $\mathcal{K}(T)$. Moreover, the interval for a particular attribute is the same for all the data records in this partition. We use $lep(a_j, i)$ and $rep(a_j, i)$ to represent the left endpoint and right

endpoint of the interval, for the attribute a_j ($1 \leq j \leq u$) in the i^{th} partition in the $\mathcal{K}(T)$, respectively. From the anonymization algorithm, we have:

$$\begin{aligned} lep(a_j, i) &= \min(a_j^{z_i}, a_j^{z_i+1}, \ldots, a_j^{z_{i+1}-1}) \\ rep(a_j, i) &= \max(a_j^{z_i}, a_j^{z_i+1}, \ldots, a_j^{z_{i+1}-1}) \end{aligned} \tag{6}$$

To find the minimum and maximum values of the set $\{ a_j^{z_i}, a_j^{z_i+1}, \ldots, a_j^{z_{i+1}-1}\}$ by the $z_{i+1} - z_i$ respondents, we employ the unconditionally secure constant-rounds *SMPC* scheme in [4]. This *SMPC* scheme provides a set of protocols that compute the shares of a function of the shared values.

Based on the result of [4], we can define a primitive comparison function $\overset{?}{<} : \mathbb{F}_\delta \times \mathbb{F}_\delta \rightarrow \mathbb{F}_\delta$ for some prime δ, such that $(\alpha \overset{?}{<} \beta) \in \{0, 1\}$ and $(\alpha \overset{?}{<} \beta) = 1$ iff $\alpha < \beta$. This function securely compares two numbers α and β, and outputs if α is less than β. With this function, the maximum and minimum numbers in a set are easily found based on a series of pairwise comparisons. We omit the details of the implementation the comparison function where the readers can find in [4].

The sub-protocol that uses the primitive comparison function $\overset{?}{<}$ to find the maximum and minimum values for the attribute a_j in the i^{th} partition is called $\mathcal{M}(a_j, i)$ with the output $< lep(a_j, i), rep(a_j, i) >$. This sub-protocol is described as follows: first, each value in this set is shared using Shamir's (t_{ss}, t_{ss}) secret sharing. The shares are distributed via an *anonymous protocol* so that the identities of the shares' owners are not revealed. Second, with the shares, the pairwise comparison of values based on $\overset{?}{<}$ can be successfully constructed. The maximum and minimum values in $\{a_j^{z_i}$, $a_j^{z_i+1}, \ldots, a_j^{z_{i+1}-1}\}$ can be found with maximally $\left\lceil \frac{3 \cdot (z_{i+1} - z_i)}{2} \right\rceil - 2$ number of pairwise comparisons. Finally, the owners of the maximum value and minimum value publish their values of a_j anonymously and each respondent in the partition assigns the values of $lep(a_j, i)$ and $rep(a_j, i)$ accordingly.

For each non-sensitive attribute a_j ($1 \leq j \leq u$) and each partition i ($1 \leq i \leq \pi$), $\mathcal{M}(a_j, i)$ is run once. Therefore, the $\mathcal{M}$ sub-protocol runs for $\pi \cdot u$ rounds. Since the $\mathcal{M}$ sub-protocol runs independently within each partition, the sub-protocol can run simultaneously for each partition. By the end, the respondent j in the i^{th} partition submits the anonymized data record $\mathcal{K}(t_j)$=$\{[lep(a_1, i), rep(a_1, i)], \ldots, [lep(a_u, i), rep(a_u, i)]$, $s_1, \ldots, s_v\}$ to the server. After collecting $\mathcal{K}(t_1), \mathcal{K}(t_2), \ldots, \mathcal{K}(t_x)$ from all x respondents, the final anonymized table $\mathcal{K}(T)$ is created and is returned to the collector.

5 Analysis

5.1 Information Leakage

Theorem 2. *The privacy preserving data collection protocol only leaks $\mathcal{L}_{svr}$ for the server and $\mathcal{L}_i$ for the respondent i, where $\mathcal{L}_{svr} = \{\mathcal{RG}_{svr}, \mathcal{F}(S)\}$ and $\mathcal{L}_i = \{\mathcal{RG}_i\}$.*

Proof. We first construct the simulator M_{svr} for the server. In step 1 in stage 1 , the knowledge of the server is described by $\mathcal{RG}_{svr}$, in which the server knows the range

of each of the random numbers in $E(R)$ and $E(R_{init})$. Each joint encrypted random number in $E(R)$ and $E(R_{init})$ in the view of the server can be simulated by M_{svr} by multiplying an encrypted random number in the range of $[\rho_{min}, \rho_{max}]$ to the encrypted random number contributed by the server. In step 2, the $E(R^+)$ is constructed based on $E(R)$ and $E(R_{init})$, where no information is leaked during the computation based on the semantic security of the Paillier's encryption. Therefore, M_{svr} simulates $E(R^+)$ based on the simulations of $E(R)$ and $E(R_{init})$. In step 3, the server gains no information about the retrieved item which is guaranteed by the property of $\mathcal{PIR}()$ function. The decrypted value in step 4 is $\mathcal{F}(S)$, which is part of the knowledge of the server. In stage 2, the input is based on $\mathcal{F}(S)$, therefore the server does not gain any additional information. In stage 3, the server receives the anonymized tuples from the respondents, the received data are equivalent to the knowledge of the server $\mathcal{K}(T)$.

Now, we construct the simulator M_i for the respondent i. In stage step 1 in stage 1, the knowledge of respondent i is described by $\mathcal{RG}_i$, in which he knows the range of joint random numbers which are jointly created by him and the server. The respondent is not participating in step 2. In step 3, M_i simulates the retrieved ciphertext by a random ciphertext. In step 4, M_i simulates the partially decrypted message by partially decrypted the random ciphertext. The respondent is not participating in stage 2. In stage 3, the secret shares and messages can be simulated by M_i using random ciphers, guaranteed by the function sharing algorithm in [4]. The output is equivalent to the knowledge of the respondent $\mathcal{K}(T)$.

5.2 γ-Concealing Property

A property explaining how well the mapped value $r_{c_i}^+$ hides the value c_i is described by the notion of γ-concealing. The value of $1-\gamma$ (the probability the adversary can guess c_i correctly based on $r_{c_i}^+$) can be approximated as follows: with $r_{c_i}^+$, the Maximum Likelihood Estimation of c_i is $c_{mle} = \left\| r_{c_i}^+/\mu \right\| - b$ (i.e. c_{mle} =roundup($r_{c_i}^+/\mu$)$-b$). As the condition for $c_i = \left\| r_{c_i}^+/\mu \right\| - b$ is equivalent to the condition for $r_{c_i}^+$ to be in the range of $[(c_i - \frac{1}{2} - b)\mu, (c_i + \frac{1}{2} - b)\mu]$, we can establish the following equivalence:

$$\Pr(c_{mle} = c_i | r_{c_i}^+) = \Pr(r_{c_i}^+ \in [(c_i - \frac{1}{2} - b)\mu, (c_i + \frac{1}{2} - b)\mu]) \tag{7}$$

The probability value on the r.h.s of the above equation can be approximated using the *central limit theorem*. According to the *central limit theorem*, $r_{c_i}^+$ is approximately normally distributed with $r_{c_i}^+ \sim N((c_i+b)\mu, (c_i+b)\sigma^2)$. Thus, the following approximation holds:

$$\begin{aligned} 1-\gamma \approx\ & \Phi_{(c_i+b)\mu,(c_i+b)\sigma^2}[(c_i + \tfrac{1}{2} - b)\mu] \\ & -\Phi_{(c_i+b)\mu,(c_i+b)\sigma^2}[(c_i - \tfrac{1}{2} - b)\mu] \end{aligned} \tag{8}$$

In the above equation, $\Phi_{(c_i+b)\mu,(c_i+b)\sigma^2}$ is the distribution function of a normal distribution with mean $(c_i+b)\mu$, and variance $(c_i+b)\sigma^2$. The equation shows that, the value of $1-\gamma$ relies on the values of μ, σ^2, b and c_i. Particularly, the protocol tends to be secure when large σ^2, b, and c_i values, and small μ value are used.

6 Experimental Evaluation

In this section, we carry out several experiments to evaluate the performance of the proposed privacy preserving data collection protocol. The experiments are divided into four parts: in the first part, we evaluate the γ-concealing property of the proposed protocol. In the second part, we evaluate the *probabilistic distance preserving* property in the proposed protocol due to its importance in utility preservation. In the third part, we evaluated the performance of the protocol in utility preservation. In order to compare with *FALL* − the k-anonymization algorithm that the proposed protocol is based on, we employ the utility metric *GCP* [6]. Lastly, we evaluate the running time of the protocol to show the practicality.

The dataset that we use for the experiments is from the website of Minnesota Population Center (*MPC*)[1], which provides census data over various locations through different time periods. For the experiments, we have extracted 1% sample USA population records with attributes *age*, *sex*, *marital status*, *occupation* and *salary* for the year 2000. The dataset contains $2,808,457$ number of data records, however, we only use a subset of these records. Among the five attributes, the age is numerical data while others are categorical data. For the categorical data, we can use taxonomy trees (e.g. [2,9]) to convert a categorical data to numerical data for generalization purposes. Among all the seven attributes, the *salary* is considered as the sensitive attribute, while others are non-sensitive and are considered as quasi-identifiers. The domain sizes for *age*, *sex*, *marital status* and *occupation* are 80, 2, 6 and 50, respectively. The programs for the experiments are implemented in Java and run on Windows XP PC with $4.00\ GB$ memory and Intel(R) Core(TM)2 Duo CPU each at $2.53\ GHz$.

6.1 Evaluation of γ-Concealing Property

In this part of experiments, we compute some real values of $1-\gamma$ with predefined parameters based on the formulas in Equation 8, to show that the proposed protocol is privacy preserving. The Figure 4(a) shows the result of how the value of $1-\gamma$ changes with the value of μ and σ^2. In the first three rows of the table, we keep the value of μ constant ($\mu = 150$) while increasing the value of σ^2. Notice that the value of $1-\gamma$ decreases with increasing σ^2. In the last three rows of the table, we keep the values of σ^2 constant ($\sigma^2 = 7,500$) instead, and increase the values of μ. Notice in this case that the value of $1-\gamma$ increases with increasing μ. In Figure 4(b), we experimented how the value of $1-\gamma$ changes with the value of b and c_i. In the first three rows of the table, we keep the value of b constant ($b = 200$) and increase the value of c_i. We find that the value of $1-\gamma$ decreases with increasing c_i. In the last three rows of the table, we keep the value of c_i constant ($c_i = 0$) and increase b. It is true that the value of $1-\gamma$ decreases with increasing b. Since the minimum c_i is 0, the last three rows of the table shows the maximum values of $1-\gamma$ under different values of b. The values of $1-\gamma$ are all below 0.1 which supports the level of privacy that a respondent can hide his quasi-identifiers with probability at least 90% in the process of data collection. For stronger privacy, we can further lower the value of $1-\gamma$, by either decreasing μ or increasing σ^2 or b.

[1] `http://www.ipums.org/`

ρ_{min}	ρ_{max}	μ	σ^2	$1-\gamma$
100	200	150	833.333	0.119235
50	250	150	3333.33	0.0597853
0	300	150	7500	0.0398776
100	400	250	7500	0.0664135
150	450	300	7500	0.0796557
200	500	350	7500	0.0928758

(a) $b = 200$, $c_i = 100$

b	c_i	$1-\gamma$
200	100	0.0796557
200	200	0.0690126
200	300	0.0617421
200	0	0.0974767
300	0	0.0796557
400	0	0.0690126

(b) $\rho_{min} = 100$, $\rho_{max} = 300$

ρ_{min}	ρ_{max}	μ	σ^2	*DPR*
100	200	150	833.333	0.999525
50	250	150	3333.33	0.999174
0	300	150	7500	0.998411
100	400	250	7500	0.999223
150	450	300	7500	0.999438
200	500	350	7500	0.999536

$b = 200$

Fig. 4. γ-Concealing and relative distance preserving

6.2 Evaluation of *Distance Preserving* Mapping

In this part of experiments, we show that the proposed mapping function $\mathcal{F}()$ can quite well preserve the *relative distance*. For this purpose, we propose the *Distance Preserving Ratio* (DPR) metric. Given a set of pre-images $\{c_1, c_2, \ldots, c_x\}$, and the set of images $\{\mathcal{F}(c_1), \mathcal{F}(c_2), \ldots, \mathcal{F}(c_x)\}$. A *relative distance preserving triple* (*RDPT*), is a combination of three pre-images $< c_{i_1}, c_{i_2}, c_{i_3} >$ whose images $< \mathcal{F}(c_{i_1}), \mathcal{F}(c_{i_2}), \mathcal{F}(c_{i_1}) >$ preserve their relative distances. The DPR is defined as follows:

$$DPR = \frac{\text{total no. of } \textit{RDPT } < c_{i_1}, c_{i_2}, c_{i_3} >}{\text{total no. of triples } C(x, 3)} \tag{9}$$

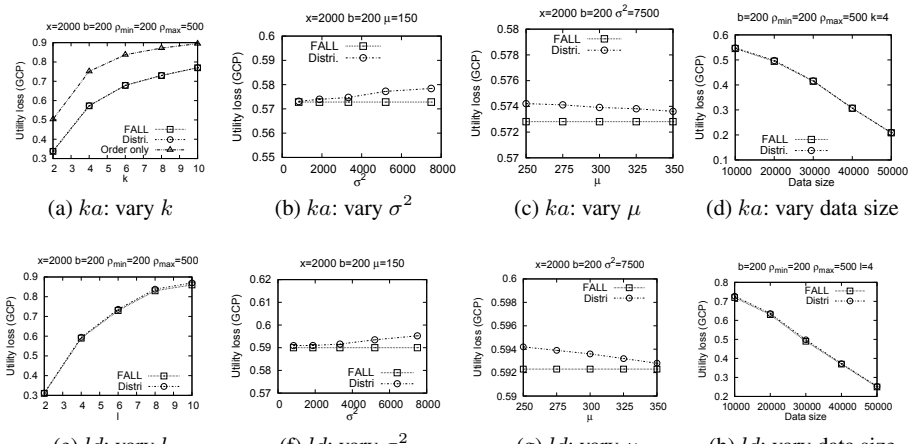
The DPR describes the ratio between the number of triples of pre-images whose mapping preserve relative distances and the total number of triples in the set of pre-images. In the experiments, we randomly select 2,000 data records from the dataset. We convert the non-sensitive attributes of selected data records into a set of integers using Hilbert curve, and input it to $\mathcal{F}()$ as the set of pre-images. The set of parameters used is the same as the one used in the experiments for γ-concealing property. In Figure 4(c), we see that when μ is fixed to 150, the value of DPR decreases with increasing of σ^2. On the other hand, when we fix the value of σ^2 to be $7,500$, the value of DPR increases with μ. In other words, large μ and small σ^2 has positive impacts on *relative distance preserving*. In all cases, the values of DPR are extremely high (almost close to 1), which clearly show that the mapping function $\mathcal{F}()$ achieves excellent *relative distance preserving*.

6.3 Evaluation of Utility Preservation

Lastly, we evaluate the utility preservation property of the proposed protocol by measuring the utility loss (the *GCP* metric) against several parameters. The set of data records used in the first three experiments is the same set of 2,000 data records used in the last part of the experiments.

In the first experiment, we measure the *GCP* value against increasing k. The parameters that we use are $b = 200$, $\rho_{min} = 200$ and $\rho_{max} = 500$. Figure 5(a) shows that the value of *GCP* increases with increasing k (as expected). Moreover, the *GCP* value computed based on table created by *FALL* (as labeled) and the proposed protocol (labeled as *Distr.*) are almost the same, showing that our approach can achieve almost the same level of utility preservation as the *FALL*. A naive method (labeled as *Order only*),

which only sorts the respondents in 1D space and group every consecutive k respondents, results in much higher *GCP* values compared to *FALL* and our approach. Figure 5(b) shows the utility loss for both *FALL* and the proposed protocol with increasing σ^2. Though from Figure 5(a), the curve of utility loss for *FALL* and the proposed protocol appear to be overlapping, when we focus the *GCP* values in the interval of [0.55, 0.6] in Figure 5(b), we indeed observe that the performance of the proposed protocol in utility preservation is slightly less optimal compare to *FALL*. Moreover, the Figure 5(b) shows that the *GCP* value based on the proposed approach increases with increasing σ^2 at relatively slow rate. Similarly, Figure 5(c) shows that increasing μ value helps to reduce the *GCP* value. In Figure 5(d), in order to evaluate how the *GCP* value changes with the data size, we increase the data size from 10,000 to 50,000. It shows that the *GCP* value for both *FALL* and the proposed approach decreases at similar rate with increasing data size. The decreasing of *GCP* value is due to the fact that when data size increases, the density of data also increases. To conclude this part, these experiments show that with appropriate parameters, the proposed approach achieves almost the same utility preservation performance as *FALL*. Figures 5(e)(f)(g)(h) show the utilities in the anonymized table based on the l-diversity heuristics in *FALL*. The pattern is similar to the experiments for k-anonymity except that the utilities for l-diversity is lower than the k-anonymity as expected.



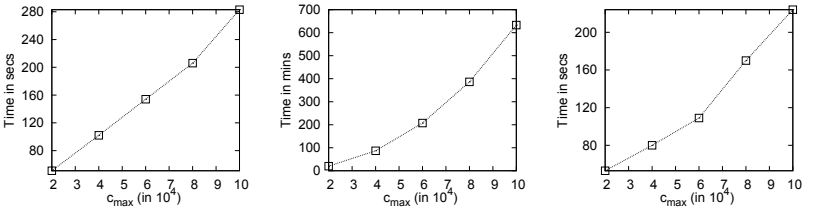
(a) ka: vary k (b) ka: vary σ^2 (c) ka: vary μ (d) ka: vary data size

(e) ld: vary l (f) ld: vary σ^2 (g) ld: vary μ (h) ld: vary data size

Fig. 5. Utility preservation evaluation

6.4 Evaluation of System Time

In this experiment, we show the practicality of our protocol using experiment implemented with java BigInteger class and Security package. We aim to verify the practicality of the solution rather than comparing its efficiency against the requirement for real time applications. The experiments results match our expectation for privacy preserving data collection application.



(a) Database creation time (b) Data preparation time (c) Query response time

Fig. 6. System time

The time evaluation focuses on the server database generation and PIR part which are the main time components of the protocol for both respondent side and server side. On the server side, the first time component is the creation of the encrypted database of size c_{max} that is equivalent to the size of domain of T as in step 1 and 2 in stage 1 of the protocol. Since c_{max} is usually large, we vary this value from 10,000 to 100,000. For the parameters setting, we choose $b = 200$, $\rho_{min} = 0$ and $\rho_{max} = 300$. In the Paillier's encryption, the modulus is set to 512 bits which creates blocks of size 512 bits on the server. Figure 6(a) shows the time (in seconds) needed by the server to create the joint encrypted random variables and encrypted database by using Paillier's encryption. Experiment shows that this step is efficient as the number of encryptions is linear to c_{max}. The second time component for the server is database preparation time. The PIR scheme in [5] requires the database to be prepared as a big integer using Chinese Remainder Theorem so as to answer PIR query. Figure 6(b) shows the database preparation time (in mins) and suggests that the time needed for this step could last for about 10.5 hours for $c_{max} = 100,000$. However, as this step is taken independently on the server for once only and requires no interaction with the data respondents, we still consider it as practical. The third server time component is the response time to the respondent's PIR query. The parameters for PIR are determined by database size and block size. In each query, the respondent privately retrieves 512 bits from the server. Figure 6(c) shows the server response time which is in minutes. Compare to the server, the client is lightly loaded in cryptographic operations. We assume 1,000 respondents are participating and measure the sum of joint random number generation time as in step 1 in stage 1 of the protocol, the query generation time and answer extraction time. Our experiments show that the PIR query generation and answer extraction at the respondent side is less sensitive to the size of database on the server side and the number of participating data respondents, and the total time needed by the respondents are less than 5 seconds. From both complexity analysis and experiment, we show that our protocol is practical. The efficiency of the our protocol could be further improved by optimizing the memory or CPU usage, or using dedicated hardware circuits for cryptographic operations.

7 Conclusions

We proposed a privacy preserving data collection protocol under the assumption that the data collector is not trustworthy. With our protocol, the collector receives an anonymized

(either k-anonymized or l-diverse) table generalized from the data records of the respondents. We show that the privacy threat caused by the information leakage remains limited. Lastly, we show with experiments that the protocol is scalable, practical and that the data utility is almost as good as in the case of a trustworthy collector.

References

1. Asmuth, C., Bloom, J.: A modular approach to key safeguarding. IEEE Trans. Information Theory 29(2), 208–210 (1983)
2. Bayardo, R., Agrawal, R.: Data privacy through optimal k-anonymization. In: Proc. of ICDE, pp. 217–228 (2005)
3. Brickell, J., Shmatikov, V.: Efficient anonymity-preserving data collection. In: KDD 2006, pp. 76–85. ACM, New York (2006)
4. Damgard, I., Fitzi, M., Kiltz, E., Nielsen, J., Toft, T.: Unconditionally secure constant-rounds multi-party computation for equality, comparison, bits and exponentiation, pp. 285–304 (2006)
5. Gentry, C., Ramzan, Z.: Single-database private information retrieval with constant communication rate, pp. 803–815 (2005)
6. Ghinita, G., Karras, P., Kalnis, P., Mamoulis, N.: Fast data anonymization with low information loss. In: Proc. of VLDB, pp. 758–769 (2007)
7. Jurczyk, P., Xiong, L.: Privacy-preserving data publishing for horizontally partitioned databases. In: CIKM 2008: Proceeding of the 17th ACM Conference on Information and Knowledge Mmanagement, pp. 1321–1322. ACM, New York (2008)
8. Kaya, K., Selçuk, A.A.: Threshold cryptography based on asmuth-bloom secret sharing. Inf. Sci. 177(19), 4148–4160 (2007)
9. LeFevre, K., DeWitt, D.J., Ramakrishnan, R.: Incognito: Efficient full-domain k-anonymity. In: Proc. of ACM SIGMOD, pp. 49–60 (2005)
10. Machanavajjhala, A., Gehrke, J., Kifer, D., Venkitasubramaniam, M.: l-diversity: Privacy beyond k-anonymity. In: Proc. of ICDE (2006)
11. Meyerson, A., Williams, R.: On the complexity of optimal k-anonymity. In: PODS 2004, pp. 223–228. ACM, New York (2004)
12. Moon, B., Jagadish, H.v., Faloutsos, C., Saltz, J.H.: Analysis of the clustering properties of the hilbert space-filling curve. IEEE TKDE 13(1), 124–141 (2001)
13. Paillier, P.: Public-key cryptosystems based on composite degree residuosity classes, pp. 223–238 (1999)
14. Samarati, P., Sweeney, L.: Generalizing data to provide anonymity when disclosing information (abstract). In: Proc. of ACM PODS, p. 188 (1998)
15. Shamir, A.: How to share a secret. Commun. ACM 22(11), 612–613 (1979)
16. Sweeney, L.: k-anonymity: A model for protecting privacy. Int. J. of Uncertainty, Fuzziness and Knowledge-Based Systems 10(5), 557–570 (2002)
17. Yang, Z., Zhong, S., Wright, R.N.: Anonymity-preserving data collection. In: KDD 2005, pp. 334–343. ACM, New York (2005)
18. Zhong, S., Yang, Z., Chen, T.: k-anonymous data collection. Inf. Sci. 179(17), 2948–2963 (2009)
19. Zhong, S., Yang, Z., Wright, R.N.: Privacy-enhancing k-anonymization of customer data. In: PODS 2005, pp. 139–147. ACM, New York (2005)