Fundamenta Informaticae 92 (2009) 411–431
DOI 10.3233/FI-2009-82
IOS Press

On Distributed k-Anonymization

Sheng Zhong
Computer Science and Engineering Department
State University of New York at Buffalo
Amherst, NY 14260, U. S. A.
szhong@cse.buffalo.edu

Abstract. When a database owner needs to disclose her data, she can k-anonymize her data to protect the involved individuals' privacy. However, if the data is distributed between two owners, then it is an open question whether the two owners can jointly k-anonymize the union of their data, such that the information suppressed in one owner's data is not revealed to the other owner. In this paper, we study this problem of *distributed* k*-anonymization*. We have two major results: First, it is impossible to design an unconditionally private protocol that implements any *normal* k-anonymization function, where normal k-anonymization functions are a very broad class of k-anonymization functions. Second, we give an efficent protocol that implements a normal k-anonymization function and show that it is private against polynomial-time adversaries. Our results have many potential applications and can be extended to three or more parties.

Keywords: k-anonymity; protocol; secure computation.

1. Introduction

Privacy concerns have been increasingly important in our society. Nowadays, it is easier than ever to locate the needed information, which gives us great convenience. But such easiness of finding information also implies easiness of violating people's privacy. To address privacy concerns, when a database containing sensitive information (e.g., health information) is made available to public access, we often eliminate the identifier attribute(s) from the database. However, only eliminating the identifier attribute is not sufficient for privacy protection, because adversaries can combine the information provided by this

Address for correspondence: Computer Science and Engineering Department, State University of New York at Buffalo, Amherst, NY 14260, U. S. A.

database with information from other public sources. A good example was given by Sweeney [31], in which one can find out who has what disease using a public database and voter lists.

Samarati and Sweeney [25] have proposed *k-anonymization*, a powerful tool to solve the above problem. We illustrate the idea of k-anonymization using a simple example: Consider a table that contains health information of patients (Table 1), where each row is a patient's phone number, age, blood test result, and urine test result. We call the set of attributes $\{\text{Phone Number}, \text{Age}\}$ a *quasi-identifier* [12, 31], because adversaries can use these attributes to identify a patient with a significant probability. In this paper, we call an attribute a *quasi-identifier attribute* if it is in the quasi-identifier. To prevent adversaries from using quasi-identifiers to violate individuals' privacy, Samarati and Sweeney suggest to make the table *k-anonymous* [25]. In a k-anonymous table, if a value of the quasi-identifier appears, then it must appear for at least k times. Therefore, each involved individual (patient in our example) is "hidden" among at least k peers, so that the adversary cannot use an individual's quasi-identifier to identify her.

The procedure of *k-anonymization* can be achieved in various ways; one possibility is that we replace some entries with $\star$ (called *suppression*). In this paper, we focus on k-anonymization by suppression.

Contact Phone Number	Age of Patient	Blood Test Result	Urine Test Result
645-3032	45	Normal	Abnormal
645-3138	25	Normal	Normal
645-3138	32	Normal	Normal
645-3138	26	Abnormal	Normal
645-3084	45	Normal	Normal

Table 1. A Table of Health Data

There have been quite a few algorithms for k-anonymization [29, 25, 31, 30, 26, 23]. Nevertheless, these algorithms can be directly applied only if the data is owned by a single entity. In some applications, the data is distributed between two (or even more) owners. For example, a survey may be carried out by two medical researchers, each of whom has a group of patients; thus each researcher owns a part of the data obtained in the survey. In such a case, ideally, the two data owners should jointly k-anonymize their data to protect the involved individuals' privacy, such that the suppressed information in each owner's data is not revealed to the other owner. We call this problem *distributed k-anonymization*. Note that distributed k-anonymization is different from *privacy-enhancing k-anonymization* [38], which performs k-anonymization while the data is collected. (See Section 2 for detailed discussion of the difference.) Our objective in this paper is to find out whether distributed k-anonymization is possible, and if possible, how we can perform distributed k-anonymization.

1.1. Our Contributions

As we have mentioned, in this paper we study the problem of distributed k-anonymization. Specifically, we assume that there is a table of data and two data owners; the table is horizontally partitioned, i.e., it is divided into two disjoint subsets of rows and each data owner has one of the subsets. We ask whether there exists a protocol that k-anonymizes the table privately. That is, we ask whether there is a protocol such that, at the end of the protocl, the two data owners output the k-anonymized table, and that each data owner learns nothing about the information suppressed in the data of the other owner. We have two major results:

- Consider the *k-anonymization function* implemented by the protocol, i.e., the function that maps each input of the protcol to the corresponding output. Our first major result is that, if we require *unconditional privacy*, the strongest form of privacy as defined in information theory, then no protocol can implement any *normal* k-anonymization function. Here normal k-anonymization functions refer to a very broad class of k-anonymization functions, including the k-anonymization functions implemented by many existing k-anonymization algorithms.

- Our second major result is that, if we relax our privacy requirement a little and require *privacy against polynomial-time adversaries*, then there is a protocol that implements a normal k-anonymization function. We give an "ID-based" protocol that does not need the two data owners to have a priori knowledge of each other's public key; as long as the two data owners know each other's ID, they can execute this protocol to k-anonymize the data privately.

We stress that our results can be extended to more parties, although in this paper we focus on two-party distributed k-anonymization only.

We briefly overview related work in Section 2. In Section 3, we present technical preliminaries. Our two major results are detailed in Sections 4 and 5, respectively. We conclude in Section 6.

2. Related Work

Now we give a very brief overview of related work from various areas.

k-Anonymization Samarati and Sweeney were the first to study k-anonymization [29, 25, 31, 30, 26]. Meyerson and Williams [23] considered k-anonymization by suppression; they showed that minimizing the number of suppressed entries is NP-hard and gave approximation algorithms for this problem. Furthermore, Aggarwal et al. [4] showed that the problem is NP-hard even when we assume ternary-valued attributes; in addition, they presented algorithms that have improved approximation ratios. Machanavajjhala et al. [22] pointed out that k-anonymity may not suffice for privacy protection in some cases; they proposed an enhanced privacy property called *ℓ-diversity*.

Zhong et al. [38] studied *privacy-enhancing k-anonymization*, which also involves distributed computing. However, their scenario is significantly different from ours: They consider two problem formulations, both involving a large number of customers, each of whom owns a row of a table; their first problem formulation is extracting the k-anonymous part of the table, not k-anonymization; in their second problem formulation, k-anonymization is performed while a miner collects the data from the customers. In contrast, in this paper we consider two owners of a distributed database; they k-anonymize the data while they combine their data together. Furthermore, the solution to the second problem formulation in [38] reveals partial information even to a polynomial-time adversary. In contrast, our protocol in Section 5 does not reveal any information to polynomial-time adversaries.

Private Two-Party Computation Chor and Kushilevitz [11] investigated private two-party computation of *boolean* functions; they showed that a boolean function has an unconditionally private protocol if and only if it is an xor of two locally computable functions. Kushilevitz [20] extended the study to general functions and showed that a general function has an unconditionally private protocol if and only if its corresponding matrix is *decomposable* (see [20] for the definition of decomposable matrix). Note that in our problem, it is non-trivial to see whether a k-anonymization function corresponds to a decomposable matrix or not. So given the result of [20], it is still open whether there are unconditionally private protocols for k-anonymization.

For privacy against polynomial-time adversaries, general-purpose cryptographic protocols have been constructed for arbitrary functions, the first of which was given by Yao [37]. (See [16] for a systematic presentation of general-purpose protocols). However, just as mentioned in [16], due to highly expensive costs in computation and communication, usually these constructions cannot be applied directly.

ID-based Cryptography In Section 5, we present a protocol private against polynomial-time adversaries, which is *ID-based*. Here being ID-based means that the involved parties do not need to have a priori knowledge of each other's public key. As long as the involved parties know each other's ID, the protocol can be executed. This gives us a lot of convenience and flexibility. Our protocol uses a specific ID-based encryption scheme developed by Waters [35], but it can be replaced by other ID-based encryption schemes with similar properties. For more about ID-based cryptography, the readers can refer to [27, 9, 8].

Other Related Work In statistical databases, there have been a good number of results on protecting individual privacy while allowing data sharing [2, 28]. The proposed methods include query restriction (e.g., [19, 10]) and data perturbation (e.g., [24, 32, 7, 1]). Also in this context, Dinur and Nissim [13] studied the tradeoff between privacy and utility.

In privacy-preserving data mining, the main technical challenge is also how to protect sensitive data while maintaining data utility. Some results in this area can be found in [6, 5, 15, 14, 18, 21, 33, 34, 17, 3].

3. Technical Preliminaries

Consider a table with m quasi-identifier attributes and m' other attributes. Without loss of generality, we assume that a_1, ..., a_m are the quasi-identifier attributes and that $a_{m+1}, \ldots, a_{m+m'}$ are the other attributes. We say the table is k-anonymous if each value of (a_1,..., a_m) either does not appear in the table, or appears in the table for at least k times.

Suppose that there are two involved parties: Alice and Bob. The table is partitioned into two (disjoint) sets of rows: Alice has N_A rows, while Bob has N_B rows. We denote Alice's part of the table by $T^{(A)}$ and Bob's part by $T^{(B)}$. The ith row of $T^{(A)}$ (resp., $T^{(B)}$) is denoted by $T_i^{(A)}$ (resp., $T_i^{(B)}$) and the jth entry of this row is denoted by $T_{i,j}^{(A)}$ (resp., $T_{i,j}^{(B)}$). Let $\mathcal{K}()$ be a function that maps $(N_A+N_B)\times(m+m')$ tables to $(N_A+N_B)\times(m+m')$ tables. We say $\mathcal{K}()$ is a *k-anonymization function* if for all $(N_A+N_B)\times(m+m')$ table T, $\mathcal{K}(T)$ is k-anonymous.

We assume that there is a private channel between Alice and Bob. By exchanging messages through this channel, Alice and Bob attempt to jointly k-anonymize the table. We say a protocol implements a

k-anonymization function $\mathcal{K}()$ if for all $T^{(A)}$, $T^{(B)}$, Alice and Bob output $\mathcal{K}(T^{(A)}, T^{(B)})$ at the end of the protocol.

Now we rigorously define our privacy requirement by adapting the standard definition of privacy in the *semi-honest model* of cryptographic protocols to our setting. In the semi-honest model, each involved party is assumed to follow the protocol but may attempt to derive extra information to violate privacy of the other party. This model has been extensively studied [16] and widely applied to privacy problems with large-size data [21, 17, 36]. Although the semi-honest model is a strong restriction on participants' behavior, there are at least two reasons for studying our problem in this model. First, deviating from the protocol requires a considerable amount of effort (to hack the computer program) and such effort is often illegal. Second, it has been shown that any protocol private in the semi-honest model can be "translated" to one secure in the fully malicious model, where the participants may deviate arbitrarily from the protocol [16].

Intuitively, our privacy requirement states that the view of the protocol seen by each party can be simulated by an algorithm that has no knowledge of the suppressed entries in the other party's data. To formalize this requirement, we must first define the *view* of each party: during an execution of the protocol, a party's view consists of this party's data, all the coin flips of this party, and all the messages this party receives. We denote by $\mathsf{view}_A(T^{(A)}, T^{(B)})$ ($\mathsf{view}_B(T^{(A)}, T^{(B)})$, resp.) the view of Alice (Bob, resp.) during an execution with the table $(T^{(A)}, T^{(B)})$.

Definition 3.1. (Unconditional Privacy) Suppose $\mathcal{K}()$ is a k-anonymization function. A distributed k-anonymization protocol that implements $\mathcal{K}()$ is *unconditionally private* if there exist two families of (randomized) algorithms $\{M_N^{(A)}\}$, $\{M_N^{(B)}\}$ such that, for any N and any $(T^{(A)}, T^{(B)})$ of size N,

$$\begin{array}{rc} & (M_N^{(A)}(T^{(A)}, \mathcal{K}(T^{(A)}, T^{(B)})), T^{(A)}, T^{(B)}) \\ \equiv & (\mathsf{view}_A(T^{(A)}, T^{(B)}), T^{(A)}, T^{(B)}), \\ & (M_N^{(B)}(T^{(B)}, \mathcal{K}(T^{(A)}, T^{(B)})), T^{(A)}, T^{(B)}) \\ \equiv & (\mathsf{view}_B(T^{(A)}, T^{(B)}), T^{(A)}, T^{(B)}), \end{array}$$

where $\equiv$ denotes equality of distributions (i.e., the random variable on the left side should be identically distributed as the random variable on the right side). The algorithms $M_N^{(A)}$ and $M_N^{(B)}$ are called *simulators* (for Alice and Bob, respectively).

Note that, in the above definition, there is no restriction on the running time of simulators; we implicitly assume that they can have unbounded computational power. Also note that the joint distribution of each simulator's output and original data must be identical to the joint distribution of the corresponding view and the original data. Privacy formalized in this way is the strongest possible, guaranteeing no extra information is revealed even to computationally unbounded adversaries. Thus we call it unconditional privacy.

Unconditional privacy is good, but not always necessary. Sometimes it suffices to guarantee privacy against polynomial-time adversaries. Consequently, we give a slightly relaxed definition of privacy as follows.

Definition 3.2. (Polynomial-time Privacy) Suppose $\mathcal{K}()$ is a k-anonymization function. A distributed k-anonymization protocol that implements $\mathcal{K}()$ is *private against polynomial-time adversaries* if there exist probabilistic polynomial-time algorithms $M^{(A)}$, $M^{(B)}$ such that, for any $(T^{(A)}, T^{(B)})$,

$$\begin{array}{rl} & \{M^{(A)}(T^{(A)}, \mathcal{K}(T^{(A)}, T^{(B)}))\}_{(T^{(A)},T^{(B)})} \\ \overset{c}{\equiv} & \{\mathsf{view}_A(T^{(A)}, T^{(B)})\}_{(T^{(A)},T^{(B)})}, \\ & \{M^{(B)}(T^{(B)}, \mathcal{K}(T^{(A)}, T^{(B)}))\}_{(T^{(A)},T^{(B)})} \\ \overset{c}{\equiv} & \{\mathsf{view}_B(T^{(A)}, T^{(B)})\}_{(T^{(A)},T^{(B)})}, \end{array}$$

where $\overset{c}{\equiv}$ denotes *computational indistinguishability* of *probability emsembles*.[1] The algorithms $M^{(A)}$ and $M^{(B)}$ are called *simulators* (for Alice and Bob, respectively).

4. Impossibility of Unconditional Privacy

In this section, we show that it is impossible to design unconditionally private protocols for a very broad class of k-anonymization functions, namely *normal* k-anonymization functions. To rigorously define normal k-anonymization functions, we first present a formal definition of suppression.

Definition 4.1. A k-anonymization function $\mathcal{K}()$ is *by suppression* if for all table T, all $i \in [1, N_A+N_B]$, all $j \in [1, m]$, either $\mathcal{K}(T)_{i,j} = T_{i,j}$ or $\mathcal{K}(T)_{i,j} = \star$, where $\star$ is a symbol not belonging to the domain of any attribute.

Normal k-anonymization functions are a class of k-anonymization functions by suppression; one important property of these functions is that they "fairly treat" the symbols of each attribute. That is to say, if we permute the symbols in any attribute's domain, the behavior of the k-anonymization function should not be affected. For example, consider two tables T and S: $T_{1,1} = 0$, $S_{1,1} = 1$; for all $i \neq 1$, $T_{i,1} = 1$, $S_{i,1} = 2$; for all $j \neq 1$, $T_{i,j} = S_{i,j}$. Clearly, if we permute the symbols in the domain of attribute a_1, mapping symbol 0 to 1 and symbol 1 to 2, then the table T becomes the table S. We hope that the small difference between T and S will not affect the behavior of the k-anonymization function. More precisely, we hope that $\mathcal{K}(T)$ also becomes $\mathcal{K}(S)$ if we permute the symbols in the domain of attribute a_1 as described above.

For simplicity, when σ is a permutation on the domain of an attribute, we use $\sigma(T)$ to denote the table obtained by applying σ to that attribute of T.

Definition 4.2. A k-anonymization function $\mathcal{K}()$ is *insensitive to permutations of symbols* if for all $j \in [1, m]$, all permutation σ on the domain of a_j,

$$\mathcal{K}(\sigma(T^{(A)}, T^{(B)})) = \sigma(\mathcal{K}(T^{(A)}, T^{(B)})).$$

Now we can formally define normal k-anonymization functions as follows.

Definition 4.3. A k-anonymization function is *normal* if the following three conditions are satisfied:

- It is a k-anonymization function by suppression;

[1] Readers can refer to, e.g., [16], for the definitions of probability ensembles and computational indistinguishability.

- It is insensitive to permutations of symbols;
- If a table is already k-anonymous, then the function maps the table to itself.

As far as we know, the k-anonymization functions implemented by many existing k-anonymization algorithms (e.g., those in [26, 23, 4]) are normal. Consequently, it is natural for us to focus on normal k-anonymization functions.

Theorem 4.1. Suppose that $\mathcal{K}()$ is a normal k-anonymization function, and that at least one of the attributes has a domain of size at least 4. If $N_A + N_B \geq 2k$, then there does not exist any unconditionally private protocol that implements $\mathcal{K}()$.

Proof:
Without loss of generality, we assume that a_1 is a domain of size at least 4, that the domain of a_1 contains symbols 0, 1, 2, and 3, and that the domains of all other attributes contain symbol 0. We define two $N_A \times (m + m')$ tables $T^{(A)}$, $S^{(A)}$, and two $N_B \times (m + m')$ table $T^{(B)}$, $S^{(B)}$ as follows:

$$T^{(A)}_{i,j} = \begin{cases} 1 & \text{if } i = 1 \wedge j = 1 \\ 0 & \text{otherwise.} \end{cases}$$

$$S^{(A)}_{i,j} = \begin{cases} 3 & \text{if } i = 1 \wedge j = 1 \\ 0 & \text{otherwise.} \end{cases}$$

$$T^{(B)}_{i,j} = \begin{cases} 2 & \text{if } i \in [1, k-1] \wedge j = 1 \\ 0 & \text{otherwise.} \end{cases}$$

$$S^{(B)}_{i,j} = \begin{cases} 3 & \text{if } i \in [1, k-1] \wedge j = 1 \\ 0 & \text{otherwise.} \end{cases}$$

Furthermore, we define σ as the permutation on the domain of a_1 such that the two symbols 1 and 3 are exchanged, but all other symbols remain unchanged. Similarly, we define π as the permutation on the domain of a_1 such that the two symbols 2 and 3 are exchanged, but all other symbols remain unchanged. Then clearly we have

$$(S^{(A)}, T^{(B)}) = \sigma(T^{(A)}, T^{(B)}); \tag{1}$$

$$(T^{(A)}, S^{(B)}) = \pi(T^{(A)}, T^{(B)}). \tag{2}$$

Since $\mathcal{K}()$ is normal, we should have

$$\mathcal{K}(\sigma(T^{(A)}, T^{(B)})) = \sigma(\mathcal{K}(T^{(A)}, T^{(B)})); \tag{3}$$

$$\mathcal{K}(\pi(T^{(A)}, T^{(B)})) = \pi(\mathcal{K}(T^{(A)}, T^{(B)})). \tag{4}$$

Plugging (1) into (3) and (2) into (4), we obtain

$$\mathcal{K}(S^{(A)}, T^{(B)}) = \sigma(\mathcal{K}(T^{(A)}, T^{(B)})); \tag{5}$$

$$\mathcal{K}(T^{(A)}, S^{(B)}) = \pi(\mathcal{K}(T^{(A)}, T^{(B)})). \tag{6}$$

Note that the only entry of 1 in $(T^{(A)}, T^{(B)})$ must have been suppressed in $\mathcal{K}(T^{(A)}, T^{(B)})$, because otherwise $\mathcal{K}(T^{(A)}, T^{(B)})$ cannot be k-anonymous. Therefore, $\mathcal{K}(T^{(A)}, T^{(B)})$ does not contain the symbol 1. Similarly, we can show that $\mathcal{K}(T^{(A)}, T^{(B)})$ does not contain the symbol 2. Furthermore, $\mathcal{K}(T^{(A)}, T^{(B)})$ does not contain the symbol 3 since $(T^{(A)}, T^{(B)})$ does not have this symbol. Hence, we obtain

$$\mathcal{K}(T^{(A)}, T^{(B)}) = \sigma(\mathcal{K}(T^{(A)}, T^{(B)})); \tag{7}$$

$$\mathcal{K}(T^{(A)}, T^{(B)}) = \pi(\mathcal{K}(T^{(A)}, T^{(B)})). \tag{8}$$

Combining (5), (6), (7), and (8), we have

$$\mathcal{K}(T^{(A)}, T^{(B)}) = \mathcal{K}(S^{(A)}, T^{(B)}) = \mathcal{K}(T^{(A)}, S^{(B)}). \tag{9}$$

Recall that $\mathcal{K}()$ maps all k-anonymous tables to themselves. Since $(S^{(A)}, S^{(B)})$ is already k-anonymous, we have $\mathcal{K}(S^{(A)}, S^{(B)}) = (S^{(A)}, S^{(B)})$, which implies

$$\mathcal{K}(S^{(A)}, S^{(B)}) \neq \mathcal{K}(T^{(A)}, T^{(B)}), \tag{10}$$

because there exists symbol $\star$ in $\mathcal{K}(T^{(A)}, T^{(B)})$.

Next, we show by contradiction that there does not exist any unconditionally private protocol that implements $\mathcal{K}()$. Suppose that there exists such an unconditionally private protocol. Then we use the techniques given by Kushilevitz [20] to show that (9) and (10) lead to a contradiction.[2]

Denote by $M(T^{(A)}, T^{(B)})$ the messages sent in the protocol when the data is $(T^{(A)}, T^{(B)})$. Since the protocol is unconditionally private, there exists an algorithm $M_N^{(B)}$ such that

$$\begin{aligned} & (M_N^{(B)}(T^{(B)}, \mathcal{K}(T^{(A)}, T^{(B)})), T^{(A)}, T^{(B)}) \\ \equiv \quad & (\mathsf{view}_B(T^{(A)}, T^{(B)}), T^{(A)}, T^{(B)}), \end{aligned}$$

which implies

$$M_N^{(B)}(T^{(B)}, \mathcal{K}(T^{(A)}, T^{(B)})) \equiv \mathsf{view}_B(T^{(A)}, T^{(B)}).$$

Note that $M(T^{(A)}, T^{(B)})$ is a part of $\mathsf{view}_B(T^{(A)}, T^{(B)})$. If we denote the rest of $\mathsf{view}_B(T^{(A)}, T^{(B)})$ by $\mathsf{view}'_B(T^{(A)}, T^{(B)})$, then we can rewrite the above equation as

$$\begin{aligned} & M_N^{(B)}(T^{(B)}, \mathcal{K}(T^{(A)}, T^{(B)})) \\ \equiv \quad & (M(T^{(A)}, T^{(B)}), \mathsf{view}'_B(T^{(A)}, T^{(B)})). \end{aligned}$$

[2] In [20], it has been shown that a function satisfying both (9) and (10) cannot have an unconditionally private protocol. However, since our definition of unconditional privacy is formalized in a slightly different way, for completeness we still include a proof based on their techniques.

Similarly, we can obtain

$$\begin{aligned} & M_N^{(B)}(T^{(B)}, \mathcal{K}(S^{(A)}, T^{(B)})) \\ \equiv \quad & (M(S^{(A)}, T^{(B)}), \mathsf{view}'_B(S^{(A)}, T^{(B)})). \end{aligned}$$

Combining (9) with the above two equations, we have

$$M(T^{(A)}, T^{(B)}) \equiv M(S^{(A)}, T^{(B)}). \tag{11}$$

Similarly, we can also obtain

$$M(T^{(A)}, T^{(B)}) \equiv M(T^{(A)}, S^{(B)}). \tag{12}$$

Now we consider a specific message sequence M_0 such that

$$\Pr[M(S^{(A)}, T^{(B)}) = M_0] > 0. \tag{13}$$

Then by (11) and (12), we know

$$\Pr[M(T^{(A)}, S^{(B)}) = M_0] > 0. \tag{14}$$

Let the number of messages in M_0 be L. Suppose that $M_0 = (M_0^{(1)}, \ldots, M_0^{(L)})$. We denote by $\Pr[M_0^{(\ell)} |T^{(A)}, (M_0^{(1)}, \ldots, M_0^{(\ell-1)})]$ (resp., $\Pr[M_0^{(\ell)} |T^{(B)}, (M_0^{(1)}, \ldots, M_0^{(\ell-1)})]$) the probability that Alice (resp., Bob) sends message $M_0^{(\ell)}$ in the ℓth round when her data is $T^{(A)}$ (resp., his data is $T^{(B)}$) and the first $\ell - 1$ rounds of messages are $(M_0^{(1)}, \ldots, M_0^{(\ell-1)})$. Thus (13) can be rewritten as

$$\begin{aligned} & \prod_{1\le\ell\le L}^{\ell \text{ is odd}} \Pr[M_0^{(\ell)}|S^{(A)}, (M_0^{(1)}, \ldots, M_0^{(\ell-1)})] \\ \times & \prod_{1\le\ell\le L}^{\ell \text{ is even}} \Pr[M_0^{(\ell)}|T^{(B)}, (M_0^{(1)}, \ldots, M_0^{(\ell-1)})] \quad > 0, \end{aligned}$$

which immediately implies

$$\prod_{1\le\ell\le L}^{\ell \text{ is odd}} \Pr[M_0^{(\ell)}|S^{(A)}, (M_0^{(1)}, \ldots, M_0^{(\ell-1)})] > 0. \tag{15}$$

Similarly, we can obtain from (14) that

$$\prod_{1\le\ell\le L}^{\ell \text{ is even}} \Pr[M_0^{(\ell)}|S^{(B)}, (M_0^{(1)}, \ldots, M_0^{(\ell-1)})] > 0. \tag{16}$$

Combining the above two equations, we have

$$\begin{aligned} & \prod_{1\le\ell\le L}^{\ell \text{ is odd}} \Pr[M_0^{(\ell)}|S^{(A)}, (M_0^{(1)}, \ldots, M_0^{(\ell-1)})] \\ \times & \prod_{1\le\ell\le L}^{\ell \text{ is even}} \Pr[M_0^{(\ell)}|S^{(B)}, (M_0^{(1)}, \ldots, M_0^{(\ell-1)})] \quad > 0, \end{aligned}$$

which is equivalent to

$$\Pr[M(S^{(A)}, S^{(B)}) = M_0] > 0. \tag{17}$$

Note that Alice's output is determined by her data and the messages in the protocol. By (13) we know that Alice outputs $\mathcal{K}(S^{(A)}, T^{(B)})$ when her data is $S^{(A)}$ and the messages are M_0. By (17) we know that Alice has the same output with a positive probability when Alice has data $S^{(A)}$ and Bob has data $S^{(B)}$. Since in this case Alice's output should always be $\mathcal{K}(S^{(A)}, S^{(B)})$, we obtain

$$\mathcal{K}(S^{(A)}, T^{(B)}) = \mathcal{K}(S^{(A)}, S^{(B)}).$$

However, (9) and (10) imply

$$\mathcal{K}(S^{(A)}, T^{(B)}) \neq \mathcal{K}(S^{(A)}, S^{(B)}).$$

Contradiction. □

5. Efficient ID-based Cryptographic Solution

Since it is impossible to design unconditionally private protocols that implement normal k-anonymization functions, a natural question is whether it is possible to design protocols private against polynomial-time adversaries for these functions. In this section, we present a protocol that implements a normal k-anonymization function and guarantees privacy against polynomial-time adversaries. An advantage of our protocol is that it is ID-based, which means the involved parties do not need to have a priori knowledge of each other's public key. Before we go into the details of our protocol design, we first explain what is ID-based cryptography, why it is useful, and what kind of ID-based encryption scheme we need in designing our protocol.

5.1. ID-based Cryptography and Waters Encryption

ID-based cryptography was first proposed by Shamir [27]; after Boneh and Franklin's seminal work [9], a lot of practical ID-based crypto-systems have been designed, for example, [8, 35]. Using an ID-based crypto-system, one can encrypt a message for any receiver without a priori knowledge of the receiver's public key. As long as the receiver's ID is known, the encryption can be easily performed. The receiver obtains a private key from the Private Key Generator (PKG); using this private key she can decrypt any message encrypted under her ID.

ID-based cryptography gives us a lot of convenience and flexibility. In particular, it makes key renewal easier because updating public key information is not necessary. For instance, one can encrypt messages under the ID "department secretary 11/2005" in November 2005, and then encrypt messages under the ID "department secretary 12/2005" in December 2005. This allows the department secretary to renew her private key every month; it is no longer necessary for the secretary to make a monthly announcement of her new public key .

In our problem of distributed k-anonymization, the involved parties are owners of large-size data. Consequently, it is essential for them to renew keys frequently. This is one of the reasons we choose to design our protocol using ID-based cryptography.

5.1.1. Waters Encryption

Our protocol is based on a specific ID-based encryption scheme, Waters encryption scheme [35], although it can be replaced by other ID-based encryption schemes with similar properties (being homomorphic and having rerandomization operations, which we shall explain shortly). To be concrete, we first give a brief review of Waters encryption scheme, which uses bilinear maps.

Definition 5.1. (Admissible Bilinear Map) A map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ (where $\mathbb{G}_1$, $\mathbb{G}_2$ are groups of the same prime order) is an *admissible bilinear map* if, for a generator g of group $\mathbb{G}_1$,

- for all a, b, $\hat{e}(g^a, g^b) = \hat{e}(g, g)^{ab}$;
- $\hat{e}(g, g) \neq 1$.

Note that admissible bilinear maps do exist: Boneh and Franklin [9] constructed such maps using Weil Pairing.

Now suppose that s is a security parameter and that q is an s-bit prime. Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be two groups of prime order q, and $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ be an admissible bilinear map with generator g of group $\mathbb{G}_1$. Assume that n is an independent parameter and that IDs are n-bit strings.

Initialization For α, β picked uniformly and independently[3] from $[0, q]$, the PKG sets $g_1 = g^\alpha$, $g_2 = g^\beta$. The PKG also chooses a random value $u' \in \mathbb{G}_1$, and a random n-dimensional vector $U = (u_1, \ldots, u_n)$, where each u_i is chosen at random from $\mathbb{G}_1$. The parameters g, g_1, g_2, u' and U are made public.

Private Key Generation For an ID v, the private key is

$$d_v = (g^{\alpha\beta}(u' \prod_{v_i=1} u_i)^r, g^r),$$

where v_i is the ith bit of v and $r \in [0, q-1]$ is picked at random.

Encryption An encryption of plaintext p under ID v is

$$C = E_v(p, t) = (p \cdot \hat{e}(g_1, g_2)^t, g^t, (u' \prod_{v_i=1} u_i)^t),$$

where $t \in [0, q-1]$ is picked at random.

Decryption Suppose that $C = (C_1, C_2, C_3)$ is a valid ciphertext under ID v. Then C can be decrypted using the private key $d_v = (d_1, d_2)$:

$$p = D_{d_v}(C) = C_1 \frac{\hat{e}(d_2, C_3)}{\hat{e}(d_1, C_2)}.$$

[3] In this paper, whenever we choose a random number, we always choose it uniformly and independently, unless otherwise specified.

Homomorphic Property The Waters ID-based encryption scheme is *homomorphic*: If we define the multiplication of two tuples as the multiplication of the corresponding elements in each dimension, then we have

$$E_v(p_1p_2, t_1 + t_2) = E_v(p_1, t_1)E_v(p_2, t_2).$$

Rerandomization Operation A ciphertext $C = (C_1, C_2, C_3)$ can be *rerandomized* using only the public parameters:

$$C' = (C_1\hat{e}(g_1, g_2)^{t'}, C_2g^{t'}, C_3(u' \prod_{v_i=1} u_i)^{t'}).$$

It can be easily verified that the result C' is another encryption of the same cleartext:

$$D_{d_v}(C') = D_{d_v}(C).$$

5.2. Solution Design

In this section, we briefly present the intuitive ideas we use to design our protocol. We leave a complete and detailed description of our protocol to Section 5.3.

5.2.1. Suppressing Less Frequent Quasi-identifiers

The first idea of our protocol design is to decide whether each quasi-identifier appears for at least k times in the table. If a quasi-identifier appears for fewer than k times, then we suppress all occurrences of this quasi-identifier.

Comparing Frequency with Threshold It is easy to decide whether a *given* quasi-identifier appears for at least k times in the table. Suppose that this quasi-identifier appears for y times in Alice's data. Then Alice prepares a list of k ciphertexts *under her own ID* to represent y: If $y < k$, the first y ciphertexts are encryptions of 1 and the remaining ciphertexts are encryptions of random numbers. If $y \geq k$, all the k ciphertexts are encryptions of 1. Alice sends this list of ciphertexts to Bob, who clearly cannot decrypt them. However, Bob can obtain a ciphertext representing whether the quasi-identifier appears for at least k times in the table: Assume that the quasi-identifier appears for y' times in Bob's data. If $y' < k$, Bob chooses the $(k - y')$th ciphertext to represent the result. Note that (with high probability) this is an encryption of 1 if and only if $k - y' \leq y$, which is equivalent to that the quasi-identifier appears for at least k times in the table. So when Bob sends this ciphertext back to Alice, all Alice needs to do is to decrypt the ciphertext and compare the cleartext with 1. If $y' \geq k$, Bob already knows that the quasi-identifier appears for at least k times in the table. So he encrypts 1 under Alice's ID and sends the ciphertext to Alice.

Multiple Quasi-identifiers Nevertheless, in our problem there are many quasi-identifiers; so we need to extend the above approach to multiple quasi-identifiers. Here the major technical challenge is that, when Alice sends many k-ciphertext lists to Bob, Bob does not know which list corresponds to which quasi-identifier. Note that Alice cannot tell Bob the quasi-identifiers corresponding to these ciphertext lists, because otherwise Bob would learn the set of quasi-identifiers appeared in Alice's data.

To solve this problem, for each list of ciphertexts sent by Alice, Bob associates the list with *every* quasi-identifier appeared in his data and applies the above described approach. Suppose that this list of ciphertexts actually corresponds to quasi-identifier x_i, which appears y_i times in Alice's data. Then for each quasi-identifier x'_j, which appears y'_j times in Bob's data, Bob uses the approach we just described above to obtain a ciphertext that represents whether $y_i + y'_j \geq k$. Then this ciphertext is an encryption of 1 if $y_i + y'_j \geq k$, and an encryption of random cleartext if $y_i + y'_j < k$. Since we are only interested in the case $x_i = x'_j$ (i.e., the case Bob's quasi-identifier matches Alice's list of ciphertext), we want to multiply this ciphertext by an encryption of random cleartext when $x_i \neq x'_j$. To achieve this goal, we note that, for a random exponent $\theta_{i,j}$, if $x_i \neq x'_j$, $(\frac{x_i}{x'_j})^{\theta_{i,j}}$ is a random element; if $x_i = x'_j$, $(\frac{x_i}{x'_j})^{\theta_{i,j}} = 1$. Thus it suffices for Bob to multiply the above ciphertext by an encryption of $(\frac{x_i}{x'_j})^{\theta_{i,j}}$. Now suppose that each ciphertext list is accompanied by an encryption of the corresponding quasi-identifier x_i under Alice's ID. Then computing the encryption of $(\frac{x_i}{x'_j})^{\theta_{i,j}}$ is easy because Bob can use the homomorphic property of the encryption scheme.

Hiding Numbers of Quasi-identifiers A subtle issue is how many lists of ciphertexts Alice should send to Bob and how many quasi-identifiers Bob should associate each list with. These cannot be the numbers of different quasi-identifiers appeared in Alice's and Bob's data, because we do not want them to reveal the numbers of different quasi-identifiers to each other. Our solution is to use a public upper bound of these numbers. To get a sufficient number of ciphertext lists, Alice should add "dumb" lists with x_is that do not match any legal quasi-identifiers. Similarly, Bob should add "dumb" x'_js.

5.2.2. Testing k-Anonymity

So far we have ignored a special case in our design—the case that fewer than k (but more than 0) rows in the table have suppressed quasi-identifiers. In this case, if we do not suppress more quasi-identifiers, then the table cannot be k-anonymous, although each unsuppressed quasi-identifier in the table appears for at least k times.

To deal with this special case, before Alice interacts with Bob to decide whether each quasi-identifier appears for at least k times in the table, she chooses σ—a set of k rows, and suppresses the quasi-identifier entries in σ. Nevertheless, this leads to another problem: We have to make sure our protocol implements a *normal* k-anonymization function; so when the original table is already k-anonymous, Alice cannot suppress the quasi-identifier entries in σ.

We solve the above problem by adding a phase of k-anonymity test to the beginning of our protocol. If Alice and Bob find that the table is already k-anonymous, they send their data to each other and output the table. Otherwise, they go into the second phase, in which Alice suppresses the quasi-identifiers in σ and then interacts with Bob to decide whether each quasi-identifier appears for at least k times in the table.

Basic Techniques of k-Anonymity Test In this first phase, Alice divides the quasi-identifiers in her data into two sets: $Q_{A,0}$, the set of quasi-identifiers appeared for at least k times in her data, and $Q_{A,1}$, the set of quasi-identifiers appeared for at most $k-1$ times (and at least one time) in her data. Similarly, Bob divides the quasi-identifiers in his data into $Q_{B,0}$ and $Q_{B,1}$. It is not hard to see that the table is

k-anonymous if and only if $Q_{A,1} - Q_{B,0} = Q_{B,1} - Q_{A,0}$ and every quasi-identifier in $Q_{A,1} - Q_{B,0}$ appears for at least k times in the entire table.

Let's temporarily assume that disclosing $Q_{A,0}$ and $Q_{B,0}$ does not violate privacy. Then Alice sends $Q_{A,0}$ to Bob and Bob sends $Q_{B,0}$ to Alice, so that Alice can compute $Q_{A,1} - Q_{B,0}$ and Bob can compute $Q_{B,1} - Q_{A,0}$. Now both Alice and Bob permute the quasi-identifiers in a specific order (e.g., in the increasing order of binary representations). So, the table is already k-anonymous if and only if for all i, the ith quasi-identifier in $Q_{A,1} - Q_{B,0}$ is equal to the ith quasi-identifier in $Q_{B,1} - Q_{A,0}$ and the number of this quasi-identifier's occurrences in the entire table is at least k.

For each i, using the technique we have described, it is easy to let Bob have a ciphertext (which is encrypted under Alice's ID) representing whether the quasi-identifier appears for at least k times in the entire table. To ensure that the two quasi-identifiers are equal, we can multiply this ciphertext by an encryption of a random power of the quotient of the two quasi-identifiers (which is obtained using the homomorphic property). The result is an encryption of 1 if the two quasi-identifiers are equal and the number of occurrences is at least k; it is an encryption of random cleartext otherwise. Given the ciphertext we obtain for each i, we can derive a single ciphertext that represents whether the table is k-anonymous: We simply take the product of the ciphertexts for all i. When this single ciphertext is sent back to Alice, Alice can learn whether the table is already k-anonymous.

In the above procedure of k-anonymity test, since we do not want to disclose the numbers of quasi-identifiers in $Q_{A,1} - Q_{B,0}$ and $Q_{B,1} - Q_{A,0}$, we again need to add "dumb" quasi-identifiers to reach a public upper bound of the quasi-identifier numbers.

Protecting $Q_{A,0}$ Now we go back to the assumption we have made: disclosing $Q_{A,0}$ and $Q_{B,0}$ does not violate privacy. Is this always true? Not necessary. $Q_{B,0}$ can be derived from the final output of our protocol because, after the k-anonymization, all quasi-identifiers in $Q_{B,0}$ still appear for at least k times in Bob's part of the data. Therefore, we can safely say that disclosing $Q_{B,0}$ to Alice does not violate privacy. However, we do not have a similar argument for disclosing $Q_{A,0}$ to Bob. Recall that we suppress the quasi-identifier entries in σ—this may affect some quasi-identifiers in $Q_{A,0}$, bringing down the numbers of their occurrences in Alice's data to less than k.

To minimize this problem, we choose σ to be the k rows in Alice's data with the least frequently appeared quasi-identifiers. Then clearly, at most one quasi-identifier in $Q_{A,0}$ can be affected. Suppose that a quasi-identifier $x^\star \in Q_{A,0}$ is affected. In this case, disclosing $Q_{A,0}$ to Bob violates privacy, but disclosing $Q'_{A,0} = Q_{A,0} - \{x^\star\}$ does not. So at the beginning of our k-anonymity test, Alice sends $Q'_{A,0}$, not $Q_{A,0}$, to Bob. In this case, it is not hard to see that the table is k-anonymous if and only if (a) for all i, the ith quasi-identifier in $Q_{A,1} - Q_{B,0}$ is equal to the ith quasi-identifier in $Q_{B,1} - Q'_{A,0}$ and the number of this quasi-identifier's occurrences in the entire table is at least k; or (b) for all i, the ith quasi-identifier in $(Q_{A,1} - Q_{B,0}) \cup \{x^\star\}$ is equal to the ith quasi-identifier in $Q_{B,1} - Q'_{A,0}$ and the number of this quasi-identifier's occurrences in the entire table is at least k.

Using the technique we have described, Bob can obtain two ciphertexts (both of which encrypted under Alice's ID), one representing whether (a) holds, and the other representing whether (b) holds. Note that (a) and (b) cannot hold simultaneously. Therefore, these two ciphertexts are one encryption of 1 and one encryption of random cleartext if the table is k-anonymous; they are both encryptions of random cleartexts otherwise. Bob switches the order of the two ciphertexts with probability $\frac{1}{2}$, so that Alice won't know learn which ciphertext corresponds to which condition. Then Bob sends them to Alice, who decrypts them to get the result of k-anonymity test.

5.2.3. Ensuring Protocol Correctness

Finally, we notice a small probability of failure using the techniques described above: We always use a ciphertext of 1 to represent “yes” and a ciphertext of random cleartext to represent “no.” Since the random cleartext can be equal to 1 with a negligible probability, the output of our protocol can be wrong with a negligible probability. Nevertheless, having a wrong output is undesirable, even if it only happens with a negligible probability. Consequently, we add a step to the end of the protocol to ensure the correctness: The two parties check whether the table they obtained is k-anonymous. If it is, they output the table; otherwise, they send their original data to each other and perform k-anonymization. Note that sending original data to each other violates privacy. However, this only happens with a negligible probability, and thus is not a problem when we consider polynomial-time adversaries.

5.3. Protocol

Below is a detailed description of our protocol.

Phase 1 Alice divides the quasi-identifiers in her data into two sets: $Q_{A,0}$, the set of quasi-identifiers appeared for at least k times in her data, and $Q_{A,1}$, the set of quasi-identifiers appeared for at most $k-1$ times but at least one time in her data. Let σ be the k rows in Alice's data with the least frequently appeared quasi-identifiers. Let $Q'_{A,0}$ be the set of quasi-identfiers that appear for at least k times in $T^{(A)} - \sigma$. Alice sends $Q'_{A,0}$ to Bob.

Bob divides the quasi-identifiers in his data into two sets: $Q_{B,0}$, the set of quasi-identifiers appeared for at least k times in his data, and $Q_{B,1}$, the set of quasi-identifiers appeared for at most $k-1$ times but at least one time in his data. Bob sends $Q_{B,0}$ to Alice.

Alice permutes the quasi-identifiers in $Q_{A,1} - Q_{B,0}$ in the increasing order of their binary representations. Suppose that these quasi-identifiers are $x_1, \ldots, x_{w_A}$, in the above mentioned order. For each x_i ($1 \leq i \leq w_A$), let y_i be the number of rows in Alice's data with quasi-identifier x_i. Then, let Λ be an element of $\mathbb{G}_1$ that does not correspond to any valid quasi-identifier. For each $i \in [w_A + 1, \max\{N_A, N_B\}]$, Alice sets $x_i = \Lambda$ and $y_i = k$. For each $i \in [1, \max\{N_A, N_B\}]$, Alice encrypts x_i under her own ID:

$$X_i^{(0)} = E_A(x_i, t_{A,i,0}),$$

where $t_{A,i,0}$ is picked uniformly and independently from $[0, q-1]$. For each $i \in [1, \max\{N_A, N_B\}]$, each $j \in [1, k]$, Alice defines

$$z_{i,j} = \begin{cases} 1 & \text{if } j \leq y_i \\ \text{random element of } \mathbb{G}_1 & \text{otherwise}. \end{cases}$$

Alice encrypts each $z_{i,j}$ under her own ID:

$$Z_{i,j}^{(0)} = E_A(z_{i,j}, t_{A,i,j}),$$

where $t_{A,i,j}$ is picked uniformly and independently from $[0, q-1]$.

If $Q'_{A,0} = Q_{A,0}$, Alice does the follows: For each $i \in [1, \max\{N_A, N_B\}]$, Alice sets $X_i^{(1)}$ to a random encryption of random cleartext; for each $i \in [1, \max\{N_A, N_B\}]$, each $j \in [1, k]$, Alice sets $Z_{i,j}^{(1)}$ to a random encryption of random cleartext.

If $Q'_{A,0} \neq Q_{A,0}$, Alice does the follows: Let the only quasi-identifier in $Q_{A,0} - Q'_{A,0}$ be $x^\star$. Suppose that the binary representation of $x^\star$ is greater than that of x_I but smaller than that of x_{I+1}. (If the binary representation of $x^\star$ is greater than that of x_{w_A}, then Alice defines $I = w_A$; if the binary representation of $x^\star$ is smaller than that of x_1, then Alice defines $I = 0$.) For each $i \in [1, I]$, Alice sets $X_i^{(1)}$ to a rerandomization of $X_i^{(0)}$; For each $i \in [1, I]$ and each $j \in [1, k]$, Alice sets $Z_{i,j}^{(1)}$ to a rerandomization of $Z_{i,j}^{(0)}$. Alice sets $X_{I+1}^{(1)}$ to a random encryption of $x^\star$. Let $y^\star$ be the number of times $x^\star$ appears in Alice's data. For each $j \in [1, y^\star]$, Alice sets $Z_{I+1,j}^{(1)}$ to a random encryption of 1; for each $j \in [y^\star + 1, k]$, Alice sets $Z_{I+1,j}^{(1)}$ to a random encryption of random cleartext. For each $i \in [I + 2, \max\{N_A, N_B\}]$, Alice sets $X_i^{(1)}$ to a rerandomization of $X_{i-1}^{(0)}$; For each $i \in [I + 2, \max\{N_A, N_B\}]$ and each $j \in [1, k]$, Alice sets $Z_{i,j}^{(1)}$ to a rerandomization of $Z_{i-1,j}^{(0)}$.

Regardless of whether $Q_{A,0} = Q'_{A,0}$ or $Q_{A,0} \neq Q'_{A,0}$, Alice sends $((X_1^{(0)}, (Z_{1,1}^{(0)}, \ldots, Z_{1,k}^{(0)})), \ldots, (X_{\max\{N_A,N_B\}}^{(0)}, (Z_{\max\{N_A,N_B\},1}^{(0)}, \ldots, Z_{\max\{N_A,N_B\},k}^{(0)})))$ and $((X_1^{(1)}, (Z_{1,1}^{(1)}, \ldots, Z_{1,k}^{(1)})), \ldots, (X_{\max\{N_A,N_B\}}^{(1)}, (Z_{\max\{N_A,N_B\},1}^{(1)}, \ldots, Z_{\max\{N_A,N_B\},k}^{(1)})))$ to Bob.

Bob permutes the quasi-identifiers in $Q_{B,1} - Q'_{A,0}$ in the increasing order of their binary representations. Suppose that these quasi-identifiers are $x'_1, \ldots, x'_{w_B}$, in the above mentioned order. For each x'_i ($1 \leq i \leq w_B$), let y'_i be the number of rows in Bob's data with quasi-identifier x'_i. For each $i \in [w_B + 1, \max\{N_A, N_B\}]$, Bob sets $x'_i = \Lambda$ and $y'_i = k$. For each $i \in [1, \max\{N_A, N_B\}]$, Bob sets

$$Z_i'^{(0)} = \begin{cases} Z_{i,k-y'_i}^{(0)} & \text{if } y'_i < k \\ E_A(1, 0) & \text{if } y'_i \geq k; \end{cases}$$

and

$$Z_i'^{(1)} = \begin{cases} Z_{i,k-y'_i}^{(1)} & \text{if } y'_i < k \\ E_A(1, 0) & \text{if } y'_i \geq k. \end{cases}$$

Then Bob computes

$$R_0 = \prod_{i=1}^{\max\{N_A,N_B\}} Z_i'^{(0)} \left(\frac{X_i^{(0)}}{E_A(x'_i, 0)}\right)^{r_{i,0}},$$

and

$$R_1 = \prod_{i=1}^{\max\{N_A,N_B\}} Z_i'^{(1)} \left(\frac{X_i^{(1)}}{E_A(x'_i, 0)}\right)^{r_{i,1}},$$

where each $r_{i,j}$ is picked independently and uniformly from $[0, q - 1]$. Bob rerandomizes R_0, R_1 and with probability $\frac{1}{2}$ switches their order. Let the result of the above operations be R'_0, R'_1. Bob sends R'_0, R'_1 to Alice.

Alice decrypts R'_0, R'_1. If one of the plaintexts obtained is 1, then Alice tells Bob that the original table has been k-anonymous and sends Bob her own data $T^{(A)}$. In this case, Bob sends his own data $T^{(B)}$ back to Alice and then they output the entire table. If both plaintexts are not equal to 1, then the protocol goes into Phase 2.

Phase 2 Recall that σ is the k rows in Alice's data with the least frequently appeared quasi-identifiers. Alice suppresses the quasi-identifiers in these k rows. Then she computes $((\overline{X}_1, (\overline{Z}_{1,1}, \ldots, \overline{Z}_{1,k})), \ldots, (\overline{X}_{\max\{N_A,N_B\}}, (\overline{Z}_{\max\{N_A,N_B\},1}, \ldots, \overline{Z}_{\max\{N_A,N_B\},k})))$ from her current data just as she computed $((X_1^{(0)}, (Z_{1,1}^{(0)}, \ldots, Z_{1,k}^{(0)})), \ldots, (X_{\max\{N_A,N_B\}}^{(0)}, (Z_{\max\{N_A,N_B\},1}^{(0)}, \ldots, Z_{\max\{N_A,N_B\},k}^{(0)})))$ from her original data. She sends $((\overline{X}_1, (\overline{Z}_{1,1}, \ldots, \overline{Z}_{1,k})), \ldots, (\overline{X}_{\max\{N_A,N_B\}}, (\overline{Z}_{\max\{N_A,N_B\},1}, \ldots, \overline{Z}_{\max\{N_A,N_B\},k})))$ to Bob.

Let v_B be the total number of different quasi-identifiers in Bob's data. Suppose that the quasi-identifiers appeared in Bob's data are $x''_1, \ldots, x''_{v_B}$. For each x''_i ($1 \leq i \leq v_B$), let y''_i be the number of rows in Bob's data with quasi-identifier x''_i. For each $i \in [v_B + 1, \max\{N_A, N_B\}]$, Bob sets $x''_i = \Lambda'$ and $y''_i = k$, where $\Lambda' \neq \Lambda$ is another element of $\mathbb{G}_1$ that does not correspond to any quasi-identifier. For each $i \in [1, \max\{N_A, N_B\}]$ and each $j \in [1, \max\{N_A, N_B\}]$, Bob computes

$$\gamma_{i,j} = \begin{cases} \overline{Z}_{i,k-y''_j}(\frac{\overline{X}_i}{E_A(x''_j,0)})^{\theta_{i,j}} & \text{if } y''_j < k \\ (\frac{\overline{X}_i}{E_A(x''_j,0)})^{\theta_{i,j}} & \text{if } y''_j \geq k, \end{cases}$$

where each $\theta_{i,j}$ is picked from $[0, q-1]$ independently and uniformly. For each $i \in [1, \max\{N_A, N_B\}]$, Bob repermutes the list $(\gamma_{i,1}, \ldots, \gamma_{i,\max\{N_A,N_B\}})$ randomly; let the result be $(\gamma'_{i,1}, \ldots, \gamma'_{i,\max\{N_A,N_B\}})$. Bob sends $(\gamma'_{i,j})_{i\in[1,\max\{N_A,N_B\}],j\in[1,\max\{N_A,N_B\}]}$ to Alice.

For each i that corresponds to a quasi-identifier in her current data, Alice decrypts $(\gamma'_{i,1}, \ldots, \gamma'_{i,\max\{N_A,N_B\}})$ to see whether there is a 1. If not, Alice suppresses all occurrences of this quasi-identifier. After suppressing all necessary quasi-identifiers, Alice sends Bob a list of quasi-identifiers that remain in her current data.

For each quasi-identifier in Bob's data, if it appears for at most $k - 1$ times and it is not in the list sent by Alice, then Bob suppresses all its occurrences.

Bob and Alice send their current data to each other and check whether the current table is k-anonymous. If it is, then they output the entire current table. Otherwise, they send their original data to each other and run a good k-anonymization algorithm on the entire original table; finally they output the result of the k-anonymization algorithm.

5.4. Protocol Analysis

Theorem 5.1. The protocol presented in Section 5.3 implements a normal k-anonymization function.

The proof of Theorem 5.1 is straightforward; so we do not present it here.

Theorem 5.2. The protocol presented in Section 5.3 is private against polynomial-time adversaries.

Proof:
We construct the two simulators as follows. (All encrypting operations below are performed under the ID of Alice.)

$M^{(A)}$ simulates Alice's data $T^{(A)}$ and coin flips using $T^{(A)}$ itself (which is one of $M^{(A)}$'s input) and coin flips of the same distributions. $M^{(A)}$ simulates the first-round message Alice received using the set of quasi-identifiers appeared for at least k times in the last N_B rows of $\mathcal{K}(T^{(A)}, T^{(B)})$. If $\mathcal{K}(T^{(A)}, T^{(B)})$ does not have any entry of $\star$, $M^{(A)}$ simulates the second-round message (R'_1, R'_2) Alice received using a random encryption of 1 and a random encryption of random cleartext, in a random order; $M^{(A)}$ simulates the third-round message ($T^{(B)}$) Alice received using the last N_B rows of $\mathcal{K}(T^{(A)}, T^{(B)})$; then $M^{(A)}$ finishes the simulation in this case. Otherwise, $M^{(A)}$ simulates the second-round message Alice received using two random encryptions of random cleartexts and proceeds to the simulation of the second phase.

At the beginning of second phase, $M^{(A)}$ takes $T^{(A)}$ and suppresses the quasi-identifiers in σ (the k rows with the least frequently appeared quasi-identifiers in $T^{(A)}$); let the result be $S^{(A)}$. Now $M^{(A)}$ considers each quasi-identifier that appears for at least one time but at most $k-1$ times in $S^{(A)}$, and at most $k-1$ times in the last N_B rows of $\mathcal{K}(T^{(A)}, T^{(B)})$. If this quasi-identifier also appears in the first N_A rows of $\mathcal{K}(T^{(A)}, T^{(B)})$, $M^{(A)}$ simulates the corresponding $(\gamma'_{i,1}, \ldots, \gamma'_{i,\max\{N_A,N_B\}})$ in the third-round message using $\max\{N_A, N_B\} - 1$ random encryptions of random cleartexts and a random encryption of 1, in a random order; otherwise, $M^{(A)}$ simulates $(\gamma'_{i,1}, \ldots, \gamma'_{i,\max\{N_A,N_B\}})$ using $\max\{N_A, N_B\}$ random encryptions of random cleartexts in a random order. $M^{(A)}$ simulates all the remaining parts of the third-round message Alice received using random encryptions of random cleartexts. $M^{(A)}$ simulates the fourth-round message Alice received using the last N_B rows in $\mathcal{K}(T^{(A)}, T^{(B)})$. $M^{(A)}$ does not simulate the possible fifth-round message Alice received, because it only appears with a negligible probability.

$M^{(B)}$ simulates Bob's data $T^{(B)}$ and coin flips using $T^{(B)}$ itself (which is one of $M^{(B)}$'s input) and coin flips of the same distributions. $M^{(B)}$ simulates the first-round message Bob received using the set of quasi-identifiers appeared for at least k times in the first N_A rows of $\mathcal{K}(T^{(A)}, T^{(B)})$. $M^{(B)}$ simulates the second-round message Bob received using $2(k+1)\max\{N_A, N_B\}$ random encryptions of random cleartexts. If $\mathcal{K}(T^{(A)}, T^{(B)})$ does not have any entry of $\star$, $M^{(B)}$ simulates the third-round message he received using "The table has been k-anonymous" and the first N_A rows in $\mathcal{K}(T^{(A)}, T^{(B)})$; then $M^{(B)}$ finishes the simulation in this case. Otherwise, $M^{(B)}$ proceeds to the simulation of the second phase.

In the second phase, $M^{(B)}$ simulates the third-round message Bob received using $(k+1)\max\{N_A, N_B\}$ random encryptions of random cleartexts. $M^{(B)}$ simulates the fourth-round message Bob received using the list of quasi-identifiers in the last N_B rows of $\mathcal{K}(T^{(A)}, T^{(B)})$. $M^{(B)}$ simulates the fifth-round message Bob received using the first N_A rows in $\mathcal{K}(T^{(A)}, T^{(B)})$. $M^{(B)}$ does not simulate the possible sixth-round message, also because it only appears with a negligible probability.

The computational indistinguishability immediately follows from the security of the encryption scheme. □

6. Conclusion and Future Work

In this paper, we investigate distributed k-anonymization between two owners of a (horizontally partitioned) distributed database. We show that unconditional privacy cannot be achieved for normal k-anonymization functions. We present an efficient ID-based protocol that implements a normal k-anonymization function and achieves privacy against polynomial-time adversaries.

While our protocol provides very strong privacy protection, we must admit that it is *not optimal* in terms of k-anonymization. For example, we note that, for each row, our protocol either suppresses its entire quasi-identifier, or keeps its quasi-identifier: it never suppresses part of a quasi-identifier. Clearly, our protocol has suppressed more than necessary to achieve k-anonymity. We leave it for future investigation how to design a privacy-preserving protocol for optimal k-anonymization.

In another aspect, our results can be extended to three or more parties. First, we observe that, given any multi-party protocol, if we divide the participants into two groups, then we obtain a two-party protocol. Consequently, given the impossibility of unconditional privacy for two parties, in the settings of three or more parties, there is no unconditionally private protocol that implements normal k-anonymization functions as well. Second, it is non-trivial to design a multi-party protocol private against polynomial-time adversaries based on our two-party protocol. However, we expect that the techniques we use in the design of our two-party protocol will also be useful in the design of multi-party protocols.

References

[1] Achugbue, J. O., Chin, F. Y.: The effectiveness of output modification by rounding for protection of statistical databases, *INFOR*, **17**(3), 1979, 209–218.

[2] Adam, N., Worthmann, J.: Security-control methods for statistical databases: a comparative study, *ACM Comput. Surv.*, **21**(4), 1989, 515–556, ISSN 0360-0300.

[3] Aggarwal, C. C., Yu, P. S.: A Condensation Approach to Privacy Preserving Data Mining, *Proc. 9th International Conference on Extending Database technology*, Springer, 2004.

[4] Aggarwal, G., Feder, T., Kenthapadi, K., Motwani, R., Panigrahy, R., Thomas, D., Zhu, A.: k-anonymity: Algorithms and Hardness, Under review, 2004.

[5] Agrawal, D., Aggarwal, C.: On the design and quantification of privacy preserving data mining algorithms, *Proc. 20th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, 2001, ISBN 1-58113-361-8.

[6] Agrawal, R., Srikant, R.: Privacy-preserving data mining, *Proc. ACM SIGMOD Conference on Management of Data*, ACM Press, May 2000, ISBN 1-581-13218-2.

[7] Beck, L. L.: A security mechanism for statistical databases, *ACM TODS*, **5**(3), September 1980, 316–338.

[8] Boneh, D., Boyen, X.: Secure identity based encryption without random oracles, *Proceedings of the Advances in Cryptology (CRYPTO 04)*, 2004.

[9] Boneh, D., Franklin, M. K.: Identity-based encryption from the weil pairing, *Proceedings of the 21st Annual International Cryptology Conference on Advances in Cryptology*, 2001.

[10] Chin, F. Y., Ozsoyoglu, G.: Auditing and inference control in statistical databases, *IEEE Trans. Sofw. Eng.*, **SE-8**(6), April 1982, 113–139.

[11] Chor, B., Kushilevitz, E.: A zero-one law for Boolean privacy, *SIAM J. Disc. Math.*, **4**, 1991, 36–47.

[12] Dalenius, T.: Finding a needle in a haystack C or identifying anonymous census record, *Journal of Official Statistics*, **2**(3), 1986, 329–336.

[13] Dinur, I., Nissim, K.: Revealing information while preserving privacy, *Proc. 22nd ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, ACM Press, 2003, ISBN 1-58113-670-6.

[14] Evfimievski, A., Gehrke, J., Srikant, R.: Limiting privacy breaches in privacy preserving data mining, *Proc. 22nd ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, ACM Press, 2003, ISBN 1-58113-670-6.

[15] Evfimievski, A., Srikant, R., Agrawal, R., Gehrke, J.: Privacy preserving mining of association rules, *Proc. Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ACM Press, 2002, ISBN 1-58113-567-X.

[16] Goldreich, O.: *Foundations of Cryptography*, vol. 2, Cambridge University Press, 2004.

[17] Kantarcioglu, M., Clifton, C.: Privacy Preserving Distributed Mining of Association Rules on Horizontally Partitioned Data, *ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, ACM, 2002.

[18] Kargupta, H., Datta, S., Wang, Q., Sivakumar, K.: On the Privacy Preserving Properties of Random Data Perturbation Techniques, *Third IEEE International Conference on Data Mining*, Florida, Nov 2003.

[19] Kleinberg, J. M., Papadimitriou, C. H., Raghavan, P.: Auditing Boolean Attributes, *Proc. of PODS*, 2000.

[20] Kushilevitz, E.: Privacy and Communication Complexity, *IEEE Symposium on Foundations of Computer Science*, 1989.

[21] Lindell, Y., Pinkas, B.: Privacy Preserving Data Mining, *Journal of Cryptology*, **15**(3), 2002, 177–206.

[22] Machanavajjhala, A., Gehrke, J., Kifer, D., Venkitasubramaniam, M.: l-Diversity: Privacy Beyond k-Anonymity, *Proceedings of ICDE 2006*, 2006.

[23] Meyerson, A., Williams, R.: On the Complexity of Optimal K-Anonymity, *Proc. 22nd ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, Paris, France, June 2004.

[24] Reiss, S.: Practical Data Swapping: The First Steps, *ACM TODS*, **9**(1), 1984, 20–37.

[25] Samarati, P., Sweeney, L.: Generalizing data to provide anonymity when disclosing information (abstract), *Proc. of the 17th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, ACM Press, 1998, ISBN 0-89791-996-3.

[26] Samarati, P., Sweeney, L.: Optimal anonymity using k-similar, a new clustering algorithm, Under review, 2003.

[27] Shamir, A.: Identity-based cryptosystems and signature schemes, *Proceedings of CRYPTO 84 on Advances in cryptology*, 1985.

[28] Shoshani, A.: Statistical databases: Characteristics, problems and some solutions, *Proc. of the eighth International Conference on Very Large Data Bases*, 1982.

[29] Sweeney, L.: Guaranteeing anonymity when sharing medical data, the datafly system, *Proc. of Journal of the American Medical Informatics Association*, 1997.

[30] Sweeney, L.: Achieving k-anonymity privacy protection using generalization and suppression, *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.*, **10**(5), 2002, 571–588, ISSN 0218-4885.

[31] Sweeney, L.: k-anonymity: a model for protecting privacy, *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.*, **10**(5), 2002, 557–570, ISSN 0218-4885.

[32] Traub, J., Yemini, Y., Wozniakowksi, H.: The statistical Security of a Statistical Database, *ACM TODS*, **9**(4), 1984, 672–679.

[33] Vaidya, J., Clifton, C.: Privacy preserving association rule mining in vertically partitioned data, *Proc. Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2002, ISBN 1-58113-567-X.

[34] Vaidya, J., Clifton, C.: Privacy-preserving k-means clustering over vertically partitioned data, *Proc. Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ACM Press, 2003, ISBN 1-58113-737-0.

[35] Waters, B.: Efficient Identity-Based Encryption Without Random Oracles, *Proceedings of Eurocrypt 2005*, 2005.

[36] Yang, Z., Zhong, S., Wright, R. N.: Privacy-Preserving Classification without Loss of Accuracy, *SDM 2005, Proceedings of the Fifth SIAM International Conference on Data Mining*, 2005.

[37] Yao, A.: How to generate and exchange secrets, *Proceedings of the 27th IEEE Symposium on Foundations of Computer Science*, IEEE, 1986.

[38] Zhong, S., Yang, Z., Wright, R. N.: Privacy Enhancing k-Anonymization of Customer Data, *PODS 2005, Proceedings of the Twenty-Fourth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Databases*, 2005.