Secure Two-Party Differentially Private Data Release for Vertically Partitioned Data

Noman Mohammed, Dima Alhadidi, Benjamin C.M. Fung, *Senior Member*, *IEEE*, and Mourad Debbabi

Abstract—Privacy-preserving data publishing addresses the problem of disclosing sensitive data when mining for useful information. Among the existing privacy models, ϵ-*differential privacy* provides one of the strongest privacy guarantees. In this paper, we address the problem of private data publishing, where different attributes for the same set of individuals are held by two parties. In particular, we present an algorithm for differentially private data release for vertically partitioned data between two parties in the semihonest adversary model. To achieve this, we first present a two-party protocol for the exponential mechanism. This protocol can be used as a subprotocol by any other algorithm that requires the exponential mechanism in a distributed setting. Furthermore, we propose a two-party algorithm that releases differentially private data in a secure way according to the definition of secure multiparty computation. Experimental results on real-life data suggest that the proposed algorithm can effectively preserve information for a data mining task.

Index Terms—Differential privacy, secure data integration, classification analysis

1 Introduction

Huge databases exist today due to the rapid advances in communication and storing systems. Each database is owned by a particular autonomous entity, for example, medical data by hospitals, income data by tax agencies, financial data by banks, and census data by statistical agencies. Moreover, the emergence of new paradigms such as cloud computing increases the amount of data distributed between multiple entities. These distributed data can be integrated to enable better data analysis for making better decisions and providing high-quality services. For example, data can be integrated to improve medical research, customer service, or homeland security. However, data integration between autonomous entities should be conducted in such a way that no more information than necessary is revealed between the participating entities. At the same time, new knowledge that results from the integration process should not be misused by adversaries to reveal sensitive information that was not available before the data integration. In this paper, we propose an algorithm to securely integrate person-specific sensitive data from two data providers, whereby the integrated data still retain the essential information for supporting data mining tasks. The following real-life scenario further illustrates the need for simultaneous data sharing and privacy preservation of person-specific sensitive data.

- *N. Mohammed is with the School of Computer Science, McGill University, 845 Sherbrook Street West, Montreal, QC H3A 0G4, Canada. E-mail: noman.mohammed@mail.mcgill.ca.*
- *D. Alhadidi, and M. Debbabi are with CIISE, Concordia University, 1455 de Maisonneuve Blvd. West, Montreal, QC H3G 1M8, Canada. E-mail: {dm_alhad, debbabi}@ciise.concordia.ca.*
- *B.C.M. Fung is with the School of Information Studies, McGill University, 845 Sherbrook Street West, Montreal, QC H3A 0G4, Canada. E-mail: ben.fung@mcgill.ca.*

Manuscript received 9 Jan. 2012; revised 23 Jan. 2013; accepted 1 May 2013; published online 16 May 2013.
For information on obtaining reprints of this article, please send e-mail to: tdsc@computer.org, and reference IEEECS Log Number TDSC-2012-01-0004. Digital Object Identifier no. 10.1109/TDSC.2013.22.

This research problem was discovered in a collaborative project with the financial industry. We generalize their problem as follows: A bank A and a loan company B have different sets of attributes about the same set of individuals identified by the common identifier attribute (*ID*), such that bank A owns $D_A(ID, Job, Balance)$, while loan company B owns $D_B(ID, Sex, Salary)$. These parties want to integrate their data to support better decision making such as loan or credit limit approvals. In addition to parties A and B, their partnered credit card company C also has access to the integrated data, so all three parties A, B, and C are data recipients of the final integrated data. Parties A and B have two concerns. First, simply joining D_A and D_B would reveal sensitive information to the other party. Second, even if D_A and D_B individually do not contain person-specific or sensitive information, the integrated data can increase the possibility of identifying the record of an individual. The next example illustrates this point.

Example 1. Party A owns the data table $D_A(ID, Job, \ldots, Class)$, while Party B owns the data table $D_B(ID, Sex, Salary, \ldots, Class)$ as shown in Table 1. Each row in the table represents the information of an individual. The attribute *Class* contains the class label Y or N, representing whether or not the loan has been approved. Both parties want to integrate their data and use the integrated data to build a classifier on the *Class* attribute. After integrating the two data tables (by matching the *ID* field), the female lawyer becomes unique and, therefore, vulnerable to be linked to sensitive information such as salary. In other words, linking attack is possible on the fields *Job* and *Sex*.

To prevent such linking attacks, Jiang and Clifton [26] and Mohammed et al. [39] have proposed algorithms that enable two parties to integrate their data satisfying the k-*anonymity* privacy model [48], [49]. The k-*anonymity* model requires that an individual should not be identifiable from a group of size smaller than k based on the quasi-identifier (QID), where QID is a set of attributes that may serve as an identifier in the data set. For example, in Table 1 *Engineer* and *Lawyer* can be generalized [48], [49] to

TABLE 1
Original Tables

Shared		Party *A*		Party *B*		
ID	Class	Job	...	Sex	Salary	...
1	N	Writer		Male	30K	
2	N	Dancer		Male	25K	
3	Y	Writer		Male	35K	
4	N	Dancer		Female	37K	
5	Y	Engineer		Female	65K	
6	Y	Engineer		Female	35K	
7	Y	Engineer		Male	30K	
8	N	Dancer		Female	44K	
9	Y	Lawyer		Male	44K	
10	Y	Lawyer		Female	44K	

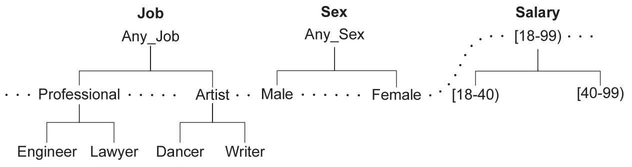


Fig. 1. Taxonomy tree for attributes Job, Sex, and Salary. The dotted line represents a “solution cut”; a concept elaborated in Section 6.

Professional according to the taxonomy presented in Fig. 1 (ignore the dotted line for now) so that this individual becomes one of many female professionals. However, Machanavajjhala et al. [34] have pointed out that with additional knowledge about the victim, k-anonymous data are vulnerable to background knowledge attacks. To prevent such attacks, *ℓ-diversity* requires that every QID group should contain at least ℓ “well-represented” values for the sensitive attribute. Similarly, there are a number of other partition-based privacy models such as *(α, k)-anonymity* [56], *(c, k)-safety* [35], and *t-closeness* [32] that differently model the adversary and have different assumptions about her background knowledge. However, recent research has indicated that these privacy models are vulnerable to various privacy attacks [54], [60], [19], [28] and provide insufficient privacy protection.

In this paper, we adopt differential privacy [14], a recently proposed privacy model that provides a provable privacy guarantee. Differential privacy is a rigorous privacy model that makes no assumption about an adversary's background knowledge. A differentially private mechanism ensures that the probability of any output (released data) is equally likely from all nearly identical input data sets and, thus, guarantees that all outputs are insensitive to any individual's data. In other words, an individual's privacy is not at risk because of the participation in the data set.

In this paper, we present an algorithm for differentially private data release for vertically partitioned data between two parties.[1] We take the single-party algorithm for differential privacy that has been recently proposed by Mohammed et al. [38] as a basis and extend it to the two-party setting. Additionally, the proposed algorithm satisfies the security definition of the semihonest adversary model. In this model, parties follow the algorithm but may try to deduce additional information from the received messages. Therefore, at any time during the execution of the algorithm, no party should learn more information about the other party's data than what is found in the final integrated table, which is differentially private. The main contribution of our paper can be summarized as follows:

- We present a two-party protocol for the exponential mechanism. We use this protocol as a subprotocol of our main algorithm, and it can also be used by any other algorithm that uses the exponential mechanism in a distributed setting.
- We present the first two-party data publishing algorithm for vertically partitioned data that generate an integrated data table satisfying differential privacy. The algorithm also satisfies the security definition in the secure multiparty computation (SMC) literature.
- We experimentally show that the differentially private integrated data table preserve information for a data mining task. In particular, taking the decision-tree induction classifier [45] as an example, we show that the proposed two-party algorithm provides similar data utility for classification analysis when compared to the single-party algorithm [38], and it performs better than the recently proposed two-party algorithm [39].

The rest of the paper is organized as follows: Section 2 presents related work. In Section 3, we present an overview of ϵ-differential privacy. In Section 4, we briefly review the security definition in the semihonest adversary model and the required cryptographic primitives. In Section 5, we describe the two-party protocol for the exponential mechanism and provide a detailed analysis of the protocol. The two-party data publishing algorithm for vertically partitioned data is presented in Section 6. In Section 7, we present the experimental results and estimate the computation and communication costs of the algorithm for a real data set. In Section 8, we answer some frequently raised questions. Finally, concluding remarks and a discussion of future work are presented in Section 9.

2 Related Work

Data privacy has been an active research topic in the statistics, database, and security communities for the last three decades [17]. The proposed methods can be roughly categorized according to two main scenarios:

- *Interactive versus noninteractive*. In an interactive framework, a data miner can pose queries through a private mechanism, and a database owner answers these queries in response. In a noninteractive framework, a database owner first anonymizes the raw data and then releases the anonymized version for data analysis. Once the data are published, the data owner has no further control over the published data. This approach is also known as privacy-preserving data publishing (PPDP) [17].
- *Single versus multiparty*. Data may be owned by a single party or by multiple parties. In the distributed (multiparty) scenario, data owners want to achieve the same tasks as single parties on their integrated data without sharing their data with others.

1. This is very different from private record linkage [24], where the goal is to identify records that represent the same real-world entity from two given data sets.

TABLE 2
Related Work

Algorithms	Data Owner			Privacy Model	
	Single	Multi		Differential Privacy	Partition-based Privacy
		Horizontally	Vertically		
LeFevre *et al.* [30], Fung *et al.* [18], etc	✓				✓
Xiao *et al.* [58], Mohammed *et al.* [38], etc.	✓			✓	
Jurczyk and Xiong [27], Mohammed *et al.* [41]		✓			✓
Jiang and Clifton [26], Mohammed *et al.* [39]			✓		✓
Our proposal			✓	✓	

Our proposed algorithm addresses the distributed and noninteractive scenario. Below, we briefly review the most relevant research works.

Single-party scenario. We have already discussed different privacy models in Section 1. Here, we provide an overview of some relevant anonymization algorithms. Many algorithms have been proposed to preserve privacy, but only a few have considered the goal for classification analysis [17]. Iyengar [25] has presented the anonymity problem for classification and proposed a genetic algorithmic solution. Bayardo and Agrawal [3] have also addressed the classification problem using the same classification metric of [25]. Fung et al. [18] have proposed a top-down specialization (TDS) approach to generalize a data table. LeFevre et al. [31] have proposed another anonymization technique for classification using multidimensional recoding [30]. More discussion about the partition-based approach can be found in the survey of Fung et al. [17].

Differential privacy [14] has recently received considerable attention as a substitute for partition-based privacy models for PPDP. However, so far most of the research on differential privacy concentrates on the interactive setting with the goal of reducing the magnitude of the added noise [11], [14], [47], releasing certain data mining results [4], [8], [9], [16], or determining the feasibility and infeasibility results of differentially-private mechanisms [5], [53], [36]. Research proposals [2], [23], [38], [58] that address the problem of noninteractive data release only consider the single-party scenario. Therefore, these techniques do not satisfy the privacy requirement of our data integration application for the financial industry. A general overview of various research works on differential privacy can be found in the survey of Dwork [12].

Distributed interactive approach. This approach is also referred to as privacy preserving distributed data mining (PPDDM) [10]. In PPDDM, multiple data owners want to compute a function based on their inputs without sharing their data with others. This function can be as simple as a count query or as complex as a data mining task such as classification, clustering, and so on. For example, multiple hospitals may want to build a data mining model for predicting disease based on patients' medical history without sharing their data with each other. In recent years, different protocols have been proposed for different data mining tasks including association rule mining [50], clustering [51], and classification [33], [6]. However, none of these methods provide any privacy guarantee on the computed output (i.e., classifier, association rules). On the other hand, Dwork et al. [13], and Narayan and Haeberlen [43] have proposed interactive algorithms to compute differentially private count queries from both horizontally and vertically partitioned data, respectively.

However, when compared to an interactive approach, a noninteractive approach gives greater flexibility because data recipients can perform their required analysis and data exploration, such as mining patterns in a specific group of records, visualizing the transactions containing a specific pattern, or trying different modeling methods and parameters.

Distributed noninteractive approach. This approach allows anonymizing data from different sources for data release without exposing the sensitive information. Jurczyk and Xiong [27] have proposed an algorithm to securely integrate horizontally partitioned data from multiple data owners without disclosing data from one party to another. Mohammed et al. [41] have proposed a distributed algorithm to integrate horizontally partitioned high-dimensional health care data. Unlike the distributed anonymization problem for vertically partitioned data studied in this paper, these methods [27], [41] propose algorithms for horizontally partitioned data.

Jiang and Clifton [26] have proposed the Distributed k-Anonymity (DkA) framework to securely integrate two data tables while satisfying the k-anonymity requirement. Mohammed et al. [39] have proposed an efficient anonymization algorithm to integrate data from multiple data owners. To the best of our knowledge, these are the only two methods [26], [39] that generate an integrated anonymous table for vertically partitioned data. However, both methods adopt k-anonymity [48], [49] or its extensions [34], [52] as the underlying privacy principle and, therefore, both are vulnerable to the recently discovered privacy attacks [54], [19], [28], [55]. Table 2 summarizes the different characteristics of the PPDP algorithms discussed above.

3 Privacy Model

Differential privacy is a recent privacy definition that provides a strong privacy guarantee. It guarantees that an adversary learns nothing more about an individual, regardless of whether her record is present or absent in the data.

Definition 3.1 (ϵ-Differential Privacy) [14]. *A randomized algorithm Ag is differentially private if for all data sets D and D', where their symmetric difference contains at most one record ($i.e., |D \triangle D'| \leq 1$), and for all possible anonymized data sets $\hat{D}$:*

$$\Pr[Ag(D) = \hat{D}] \leq e^{\epsilon} \times \Pr[Ag(D') = \hat{D}], \quad (1)$$

where the probabilities are over the randomness of Ag.

A standard mechanism to achieve differential privacy is to add a random noise to the true output of a function. The noise is calibrated according to the *sensitivity* of the function. The sensitivity of a function is the maximum difference of its outputs from two data sets that differ only in one record.

Definition 3.2 (Sensitivity) [14]. *For any function $f : D \to \mathbb{R}^d$, the sensitivity of f is*

$$\Delta f = \max_{D,D'} \|f(D) - f(D')\|_1, \tag{2}$$

for all D, D' differing in at most one record.

For example, let f be the count function. The Δf is 1 because $f(D)$ can differ at most by 1 due to the addition or removal of a single record.

Dwork et al. [14] have proposed the Laplace mechanism. The mechanism takes a data set D, a function f, and the parameter λ that determines the magnitude of noise as inputs. It first computes the true output $f(D)$, and then perturbs the output by adding noise. The noise is generated according to a Laplace distribution with probability density function $\Pr(x|\lambda) = \frac{1}{2\lambda}\mathtt{exp}(-|x|/\lambda)$; its variance is $2\lambda^2$ and its mean is 0. The Laplace mechanism guarantees that perturbed output $f(\hat{D}) = f(D) + \mathtt{Lap}(\Delta f/\epsilon)$ satisfies ϵ-differential privacy, where $\mathtt{Lap}(\Delta f/\epsilon)$ is a random variable sampled from the Laplace distribution.

McSherry and Talwar [37] have proposed the exponential mechanism to achieve differential privacy whenever it makes no sense to add noise to outputs. The exponential mechanism can choose an output $t \in \mathcal{T}$ that is close to the optimum with respect to a utility function while preserving differential privacy. It takes as inputs a data set D, an output range $\mathcal{T}$, a privacy parameter ϵ, and a utility function $u : (D \times \mathcal{T}) \to \mathbb{R}$ that assigns a real valued score to every output $t \in \mathcal{T}$, where a higher score means better data utility. In this paper, we measure the data utility in terms of classification accuracy (CA). The mechanism induces a probability distribution over the range $\mathcal{T}$ and then samples an output t. Let $\Delta u = \max_{\forall t,D,D'}|u(D,t) - u(D',t)|$ be the sensitivity of the utility function. The probability associated with each output is proportional to $\mathtt{exp}(\frac{\epsilon u(D,t)}{2\Delta u})$; that is, the output with a higher score is exponentially more likely to be chosen.

4 Security Model

In this section, we briefly present the security definition in the semihonest adversary model. Additionally, we introduce the required cryptographic primitives that are instrumented inside the proposed algorithm in this paper.

4.1 Secure Multiparty Computation

In the following, we present the security definition in the semi-honest adversary model according to Goldreich [21]:

Definition 4.1 (Security with respect to semihonest bahvior) [21]. *Let $f : \{0,1\}^* \times \{0,1\}^* \mapsto \{0,1\}^* \times \{0,1\}^*$ be a probabilistic polynomial-time functionality, where $f_1(x,y)$ ($f_2(x,y)$, respectively) denotes the first (second, respectively) element of $f(x,y)$. Let Π be a two-party protocol for computing f. Let the view of the first (second, respectively) party during an execution of protocol Π on (x,y) denoted $view_1^\Pi$ ($view_2^\Pi$, respectively) be $(x, r_1, m_1, \ldots, m_t)$ ($(y, r_2, m_1, \ldots, m_t)$, respectively), where r_1 represents the outcome of the first (r_2 the second, respectively) party's internal coin tosses and m_i represents the ith message the first (second, respectively) party has received. The output of the first (second, respectively) party during an execution of Π on (x,y) denoted $output_1^\Pi(x,y)$ ($output_2^\Pi(x,y)$, respectively) is implicit in the party's view of the execution. We say that Π securely computes f if there exist probabilistic polynomial time algorithms denoted S_1 and S_2 such that*

$$\begin{aligned}&\{(S_1(x, f_1(x,y)), f_2(x,y))\}_{x,y\in\{0,1\}} \\ &\quad \overset{c}{\equiv} \left\{\left(view_1^\Pi(x,y), output_2^\Pi(x,y)\right)\right\}_{x,y\in\{0,1\}^*} \\ &\{(f_1(x,y), S_2(x, f_1(x,y)))\}_{x,y\in\{0,1\}^*} \\ &\quad \overset{c}{\equiv} \left\{\left(output_1^\Pi(x,y), view_2^\Pi(x,y)\right)\right\}_{x,y\in\{0,1\}^*},\end{aligned}$$

where $\overset{c}{\equiv}$ denotes computational indistinguishability.

Two probability distributions are computationally indistinguishable if no efficient algorithm can tell them apart. Namely, the output distribution of every efficient algorithm is oblivious whether the input is taken from the first distribution or from the second distribution [20]. Many of the protocols, as in the case of the proposed algorithm in this paper, involve the composition of secure subprotocols in which all intermediate outputs from one subprotocol are inputs to the next subprotocol. These intermediate outputs are either simulated given the final output and the local input for each party or computed as random shares. Random shares are meaningless information by themselves. However, shares can be combined to reconstruct the result. Using the composition theorem [21], it can be shown that if each subprotocol is secure, then the resulting composition is also secure.

4.2 Cryptographic Primitives

We now list all the required cryptographic primitives.

Yao's Protocol [59]. It is a constant-round protocol for secure computation of any probabilistic polynomial-time function in the semihonest model. Let us assume that we have two parties, P_1 and P_2, with their inputs x and y, respectively. Both parties want to compute the value of the function $f(x,y)$. Then, P_1 needs to send P_2 an encrypted circuit computing $f(x,.)$. The received circuit is encrypted and accordingly P_2 learns nothing from this step. Afterwards, P_2 computes the output $f(x,y)$ by decrypting the circuit. This can be achieved by having P_2 obtaining a series of keys corresponding to its input y from P_1 such that the function $f(x,y)$ can be computed given these keys and the encrypted circuit. However, P_2 must obtain these keys from P_1 without revealing any information about y. This is done by using the oblivious transfer protocol [21].

Random Value Protocol (RVP) [7]. This protocol allows two parties to generate a random value $R \in \mathbb{Z}_Q$, where R has been chosen uniformly and $Q \in \mathbb{Z}_N$ is not known by either party but it is shared between them. More specifically, P_1 has $R_1 \in \mathbb{Z}_N$, and P_2 has $R_2 \in \mathbb{Z}_N$ such that $R = R_1 + R_2 \ mod \ N \in [0, Q-1]$, where N is the public key for the additive homomorphic scheme utilized in this protocol, namely, Paillier's scheme [44].

Secure Scalar Product Protocol (SSPP) [26], [57]. It securely computes the scalar product of two binary vectors $Z_1 = (a_1, \ldots, a_n)$ and $Z_2 = (b_1, \ldots, b_n)$ owned by two parties P_1 and P_2, respectively. At the end of this protocol, P_1 and P_2 have random shares of the result.

5 Two-Party Protocol for Exponential Mechanism

In this section, we present a two-party protocol for the exponential mechanism together with a detailed analysis. As discussed in Section 3, the exponential mechanism chooses a candidate that is close to optimum with respect to a utility function while preserving differential privacy. In the distributed setting, the candidates are owned by two parties and, therefore, a secure mechanism is required to compute the same output while ensuring that no extra information is leaked to any party.

5.1 Distributed Exponential Mechanism (DistExp)

The *distributed exponential mechanism* presented in Algorithm 1 takes the following items as input:

- Finite discrete alternatives $\langle (v_1, u_1), \ldots, (v_n, u_n) \rangle$, where a pair (v_i, u_i) is composed of the candidate v_i and its score u_i. Parties P_1 and P_2 own $(v_1, u_1), \ldots, (v_j, u_j)$ and $(v_{j+1}, u_{j+1}) \ldots (v_n, u_n)$, respectively.
- Privacy budget ϵ.

Algorithm 1. Distributed Exponential Mechanism.
Input: Candidate-score pairs $\langle (v_1, u_1), \ldots, (v_n, u_n) \rangle$ owned by the parties, and the privacy budget ϵ
Output: Winner w

```
1:  $P_1$ evaluates $s_1 \leftarrow \sum_{i=1}^{j} \mathtt{exp}(\frac{\epsilon u_i}{2\Delta \mathrm{u}})$;
2:  $P_2$ evaluates $s_2 \leftarrow \sum_{i=j+1}^{n} \mathtt{exp}(\frac{\epsilon u_i}{2\Delta \mathrm{u}})$;
3:  $P_1$ and $P_2$ execute RVP to compute random shares
      $R_1$ and $R_2$, where $(R_1 + R_2) \in _{(S_1+S_2)}$;
4:  for $k = 1$ to $n$ do
5:      if $k \leq j$ then
6:          $P_1$ evaluates $L_1 \leftarrow \sum_{i=1}^{k} \mathtt{exp}(\frac{\epsilon u_i}{2\Delta \mathrm{u}})$;
7:          $P_2$ evaluates $L_2 \leftarrow 0$;
8:      else
9:          $P_1$ evaluates $L_1 \leftarrow \sum_{i=1}^{j} \mathtt{exp}(\frac{\epsilon u_i}{2\Delta \mathrm{u}})$;
10:         $P_2$ evaluates $L_2 \leftarrow \sum_{i=j+1}^{k} \mathtt{exp}(\frac{\epsilon u_i}{2\Delta \mathrm{u}})$;
11:     end if
12:     $P_1$ and $P_2$ execute COMPARISON$(R_1, R_2, L_1, L_2)$;
13:     if $b = 0$ then
14:         $w \leftarrow v_k$;
15:         return $w$;
16:     end if
17: end for
```

Algorithm 2. COMPARISON.
Input: Random shares R_1 and R_2, and values L_1 and L_2
Output: b

```
1: $R =$ add$(R_1, R_2)$;
2: $L =$ add$(L_1, L_2)$;
3: $b =$ compare$(R, L)$;
4: return $b$;
```

The protocol outputs a winner candidate depending on its score using the exponential mechanism. The scores of the candidates can be calculated using different utility functions [38]. Given the scores of all the candidates, exponential mechanism selects the candidate v_j with the following probability, where Δu is the sensitivity of the chosen utility function:

$$\frac{\mathtt{exp}(\frac{\epsilon u_j}{2\Delta \mathrm{u}})}{\sum_{i=1}^{n} \mathtt{exp}(\frac{\epsilon u_i}{2\Delta \mathrm{u}})}. \tag{3}$$

The distributed exponential mechanism can be summarized as follows:

Computing (3). A simple implementation of the exponential mechanism is to have the interval [0, 1] partitioned into segments according to the probability mass defined in (3) for the candidates. Next, we sample a random number uniformly in the range [0, 1] and the partition in which the random number falls determines the winner candidate. However, this method involves computing a secure division (3). Unfortunately, we are not aware of any secure division scheme that fits our scenario, where the numerator value is less than the denominator value.

Alternatively, we solve this problem without a secure division protocol. We first partition the interval $[0, \sum_{i=1}^{n} \mathtt{exp}(\frac{\epsilon u_i}{2\Delta \mathrm{u}})]$ into n segments, where each segment corresponds to a candidate v_i and has a subinterval of length equal to $\mathtt{exp}(\frac{\epsilon u_i}{2\Delta \mathrm{u}})$. We then sample a random number uniformly in the range $[0, \sum_{i=1}^{n} \mathtt{exp}(\frac{\epsilon u_i}{2\Delta \mathrm{u}})]$ and the segment in which the random number falls determines the winner candidate.

Picking a Random Number R. Each party first computes individually $\mathtt{exp}(\frac{\epsilon u_i}{2\Delta \mathrm{u}})$ for its candidates. Then, both P_1 and P_2 compute $s_1 = \sum_{i=1}^{j} \mathtt{exp}(\frac{\epsilon u_i}{2\Delta \mathrm{u}})$ and $s_2 = \sum_{i=j+1}^{n} \mathtt{exp}(\frac{\epsilon u_i}{2\Delta \mathrm{u}})$, respectively. P_1 and P_2 need to pick a random number uniformly in the range $[0, s_1 + s_2]$, where $s_1 + s_2 = \sum_{k=1}^{n} \mathtt{exp}(\frac{\epsilon u_k}{2\Delta \mathrm{u}})$. This can be achieved by using the random value protocol [7]. RVP takes s_1 and s_2 from the parties as input and outputs the random value shares R_1 and R_2 to the respective parties, where $R = R_1 + R_2$. However, RVP works only in an integer setting but s_1 and s_2 can be decimal numbers because of the exponential function exp. In this case, scaling is needed and consequently the accuracy of the exponential mechanism could be slightly affected unless the scaling factor is very large. However, if the scaling factor is very large the total cost in terms of bits will increase. We experimentally measure the impact of scaling in Section 7.

We address the scaling issue by taking the floor value of $\mathtt{exp}(\frac{\epsilon u_i}{2\Delta \mathrm{u}}) \times 10^l$. Here, l is a predefined number between the parties that indicates the number of the considered digits after the decimal point. For example, the value 2.718281828 of $\mathtt{exp}(\frac{\epsilon u_i}{2\Delta \mathrm{u}})$ can be scaled in different ways according to the considered digits after the decimal point, as shown in Table 3. The parties should agree on a specific value for l and only consider the integer portion using the floor function. The higher accuracy (in terms of the number of the considered digits after the decimal point) we demand, the higher cost we pay (in terms of bits), as also shown in Table 3. These extra bits result in additional computation and communication costs. More details are

TABLE 3
Cost Analysis

l	Scaling	Floor Value	Cost (Extra Bits)
1	2.718281828×10^1	27	$log_2 10^1$
2	2.718281828×10^2	271	$log_2 10^2$
3	2.718281828×10^3	2718	$log_2 10^3$
4	2.718281828×10^4	27182	$log_2 10^4$

provided in Section 7. Notice that restricting the values of $\mathtt{exp}(\frac{\epsilon u_i}{2\Delta \mathrm{u}})$ to a finite range is completely natural as calculations performed on computers are handled in this manner due to memory constraints.

Example 2. Suppose P_1 has two candidates and the values of $\mathtt{exp}(\frac{\epsilon u_i}{2\Delta \mathrm{u}})$ for these candidates are 54.59815003 and 403.4287935, respectively; P_2 has one candidate with a value of 7.389056099. After deciding that the value of l is one and considering the floor value, P_1 ends up with the integer values 545 and 4,034, whereas P_2 ends up with the value 73. Both parties then pick a random number in the range [0, 4,652] using the RVP where $4{,}652 = 545 + 4{,}034 + 73$. Similarly, if the parties decide that the value of l is two, P_1 ends up with the integer values 5,459 and 40,342, whereas P_2 ends up with the value 738. The two parties then pick a random number in the range [0, 46,539] using the RVP, where $46{,}539 = 5{,}459 + 40{,}342 + 738$.

Picking a winner. The two parties engage in a simple secure circuit evaluation process using Yao's Protocol [59] in Line 12. The circuit COMPARISON compares their random number R with the sum $(L_1 + L_2)$ provided by P_1 and P_2, respectively. The winner v_i is the first candidate such that $R \leq L_1 + L_2$, where

$$L_1 = \sum_{i=1}^{j} \mathtt{exp}\Big(\frac{\epsilon u_i}{2\Delta \mathrm{u}}\Big) \text{ and } L_2 = 0, \text{or}$$

$$L_1 = s_1 \text{ and } L_2 = \sum_{i=j+1}^{n} \mathtt{exp}\Big(\frac{\epsilon u_i}{2\Delta \mathrm{u}}\Big).$$

Example 3 (Continued from Example 2). Suppose the two parties pick a random number in [0, 4,652] using RVP. The circuit first checks if the random number is less than or equal to 545. If so, the first candidate of P_1 is the winner; otherwise, the circuit checks again if the random number is less than or equal to $4{,}579$ $(545 + 4{,}034)$. If so, the second candidate of P_1 is the winner; otherwise, the candidate of P_2 is the winner because the random number must be within the range $[0, 4{,}652]$ according to RVP [7].

Remark. The proposed distributed exponential mechanism takes (candidate, score) pairs as inputs. The score is calculated using a utility function. The proposed distributed exponential mechanism is, therefore, independent of the choice of the utility function. In the case of vertically partitioned data, we can use two types of utility functions: 1) utility functions such as information gain, maximum function, and the widest (normalized) range of values that can be calculated locally by each party or 2) utility functions that cannot be computed locally. In the latter case, secure function evaluation techniques can be used by the parties to compute these utility functions. Once the scores of the candidates are computed using the utility functions in either case, they are ready to be used as inputs to execute the distributed exponential mechanism.

5.2 Analysis

In this section, we first prove that Algorithm 1 correctly implements the exponential mechanism. Then, we analyze the security and the efficiency of the algorithm.

Proposition 5.1 (Correctness). *Assuming both parties are semihonest, Algorithm 1 correctly implements the exponential mechanism for two parties.*

Proof. Algorithm 1 selects a candidate v_i with probability $\propto \mathtt{exp}(\frac{\epsilon u_i}{2\Delta \mathrm{u}}))$. Each party computes $\mathtt{exp}(\frac{\epsilon u_i}{2\Delta \mathrm{u}})$ for its candidates. Then, parties build an interval in the range $[0, \sum_{k=1}^{n} \mathtt{exp}(\frac{\epsilon u_k}{2\Delta \mathrm{u}})]$ and partition it among the candidates, where each subinterval has a length equal to $\mathtt{exp}(\frac{\epsilon u_i}{2\Delta \mathrm{u}})$. Since the random value lies uniformly between $[0, \sum_{k=1}^{n} \mathtt{exp}(\frac{\epsilon u_k}{2\Delta \mathrm{u}})]$ and a candidate is chosen according to this value, the probability of choosing any candidate is

$$\frac{\mathtt{exp}\big(\frac{\epsilon u_i}{2\Delta \mathrm{u}}\big)}{\sum_{k=1}^{n} \mathtt{exp}\big(\frac{\epsilon u_k}{2\Delta \mathrm{u}}\big)}.$$

Therefore, according to [37], Algorithm 1 correctly implements the exponential mechanism. □

Proposition 5.2 (Security). *Algorithm 1 is secure under the semihonest adversary model.*

Proof. The communication between P_1 and P_2 takes place in the random value protocol and in the COMPARISON circuit. Algorithm 1 is secure if both the RVP and the COMPARISON circuit are secure due to the composition theorem [21]. Since RVP [7] and COMPARISON [21] have been proven to be secure, Algorithm 1 is also secure. □

Proposition 5.3 (Complexity). *The encryption and the communication costs of Algorithm 1 are bounded by $O(nlogC)$ and $O(nKlogC)$, respectively.*

Proof. In Line 3, both parties run RVP where $O(\xi)$ and $O(\zeta)$ are the encryption and the communication costs of RVP, respectively. The add and the compare circuits determine the complexity of the COMPARISON circuit. Since the number of gates for add and compare circuits is linear to their input size, the protocol COMPARISON requires evaluation of $O(logC)$ gates, where $C = \sum_{i=1}^{n} \lfloor \mathtt{exp}(\frac{\epsilon' u_i}{2\Delta \mathrm{u}}) \times 10^l \rfloor$. Hence, the number of the encryptions and the communication complexity of COMPARISON are bounded by $O(logC)$ and $O(KlogC)$, respectively, where K is the size of the encryption and the decryption keys [22]. The COMPARISON protocol is called at most n times in Line 12. Therefore, the encryption and the communication costs are bounded by $O(\xi + nlogC)$ and $O(\zeta + nKlogC)$, respectively. Assuming, $nlogC \gg \xi$ and $nKlogC \gg \zeta$, the total encryption and communication costs of Algorithm 1 are bounded by $O(nlogC)$ and $O(nKlogC)$, respectively. □

6 Two-Party Differentially Private Data Release Algorithm

In this section, we first define some notations, state the problem, and present our assumptions. We then describe the two-party algorithm for differentially private data release for vertically partitioned data.

6.1 Preliminaries

Suppose two parties P_1 and P_2 own data table D_1 and D_2, respectively. Both parties want to release an integrated anonymous data table $\hat{D}(A_1^{pr}, \ldots, A_d^{pr}, A^{cls})$ to the public for classification analysis. The attributes in D_1 and D_2 are classified into three categories: 1) An explicit identifier attribute A^i that explicitly identifies an individual, such as *SSN* and *Name*. These attributes are removed before releasing the data. 2) A class attribute A^{cls} that contains the class value, and the goal of the data miner is to build a classifier to accurately predict the value of this attribute. 3) A set of predictor attributes $\mathcal{A}^{pr} = \{A_1^{pr}, \ldots, A_d^{pr}\}$, whose values are used to predict the class attribute. The explicit identifier and the class attribute are shared among the two parties.

Given a table D_1 owned by P_1, a table D_2 owned by P_2 and a privacy parameter ϵ, our objective is to generate an integrated anonymous data table $\hat{D}$ such that 1) $\hat{D}$ satisfies ϵ-differential privacy and 2) the algorithm to generate $\hat{D}$ satisfies the security definition of the semi-honest adversary model.

We require the class attribute to be categorical. However, the values of a predictor attribute can be either numerical v_n or categorical v_c. Further, we require that for each categorical-predictor attribute A_i^{pr}, a taxonomy tree is provided. We assume that there is no trusted third party who computes the output table $\hat{D}$ and the parties are semihonest. We also require that both the private tables D_1 and D_2 contain the same set of records (individuals), where each party holds different set of attributes. This can be achieved by executing a secure set intersection protocol on the explicit identifiers (ID) (see [39, Section 2.2] for details). Therefore, private data tables need to be preprocessed, if needed, to identify the common records. The proposed algorithm uses the common records to generate an integrated anonymous data table.

6.2 Two-Party Algorithm

In this section, we present our Distributed Differentially private anonymization algorithm based on Generalization *(DistDiffGen)* for two parties as shown in Algorithm 3. The algorithm first generalizes the raw data and then adds noise to achieve ϵ-differential privacy.

Algorithm 3. Two-Party Algorithm (DistDiffGen).
Input: Raw data set D_1, privacy budget ϵ, and number of specializations h
Output: Anonymized data set $\hat{D}$

```
1: Initialize $D_g$ with one record containing top
   most values;
2: Initialize $Cut_i$ to include the topmost value;
3: $\epsilon' \leftarrow \frac{\epsilon}{2(|A_n^{pr}|+2h)}$;
4: Determine the split value for each $v^n \in \cup Cut_i$ with
   probability $\propto \texttt{exp}(\frac{\epsilon'}{2\Delta \texttt{u}}\texttt{u}(D, v^n))$;
5: Compute the score $\forall v \in \cup Cut_i$;
6: for $l = 1$ to $h$ do
7:     Determine the winner candidate $w$ by
       Algorithm 1 (DistExp);
8:     if $w$ is local then
9:         Specialize $w$ on $D_g$;
10:        Replace $w$ with $child(w)$ in the local copy of $\cup Cut_i$;
11:        Instruct $P_2$ to specialize and update $\cup Cut_i$;
12:        Determine the split value for each new $v^n \in \cup Cut_i$
           with probability $\propto \texttt{exp}(\frac{\epsilon'}{2\Delta \texttt{u}}\texttt{u}(D, v^n))$;
13:        Compute the score for each new $v \in \cup Cut_i$;
14:    else
15:        Wait for the instruction from $P_2$;
16:        Specialize $w$ and update $\cup Cut_i$ using
           the instruction;
17:    end if
18: end for
19: for each leaf node of $D_g$ do
20:    Execute the SSPP Protocol to compute the shares $C_1$
       and $C_2$ of the true count $C$;
21:    Generate two gaussian random variables
       $Y_i \sim \mathcal{N}(0, \sqrt{1/\epsilon})$ for $i \in \{1, 2\}$;
22:    Compute $X_1 = C_1 + Y_1^2 - Y_2^2$;
23:    Exchange $X_1$ with $P_2$ to compute $(C + \texttt{Lap}(2/\epsilon))$;
24: end for
25: return Each leaf node with count $(C + \texttt{Lap}(2/\epsilon))$
```

The general idea is to anonymize the raw data by a sequence of specializations starting from the topmost general state. A specialization, written $v \rightarrow child(v)$, where $child(v)$ denotes the set of child values of v, replaces the parent value v with child values. The specialization process can be viewed as pushing the "cut" of each taxonomy tree downwards. A *cut* of the taxonomy tree for an attribute A_i^{pr}, denoted by Cut_i, contains exactly one value on each root-to-leaf path. Fig. 1 shows a solution cut indicated by the dotted line. The specialization starts from the topmost cut and pushes down the cut iteratively by specializing a value in the current cut.

Algorithm 3 is executed by the party P_1 (same for the party P_2) and can be summarized as follows:

Generalizing raw data. Each party keeps a copy of the current $\cup Cut_i$ and a generalized table D_g as shown in Fig. 2, in addition to the private table D_1 or D_2. Here, $\cup Cut_i$ is the set of all candidate values for specialization. Initially, all values in $\mathcal{A}^{pr}$ are generalized to the topmost value in their taxonomy trees (Line 1), and Cut_i contains the topmost value for each attribute A_i^{pr} (Line 2). At each iteration, the algorithm uses the distributed exponential mechanism (Algorithm 1) to select a candidate $w \in \cup Cut_i$, which is owned by either P_1 or P_2, for specialization (Line 7).

Candidates are selected based on their score values, and different utility functions can be used to determine the scores of the candidates. Once a winner candidate is determined, both parties specialize the winner w on D_g by splitting their records into child partitions according to the provided taxonomy trees. If the winner w is one of P_1's candidates, P_1 specializes w on D_g (Line 9), updates its local copy of $\cup Cut_i$ (Line 10), and instructs P_2 to specialize and update its local copy of $\cup Cut_i$ accordingly (Line 11). P_1 also calculates the scores of the new candidates due to the specialization

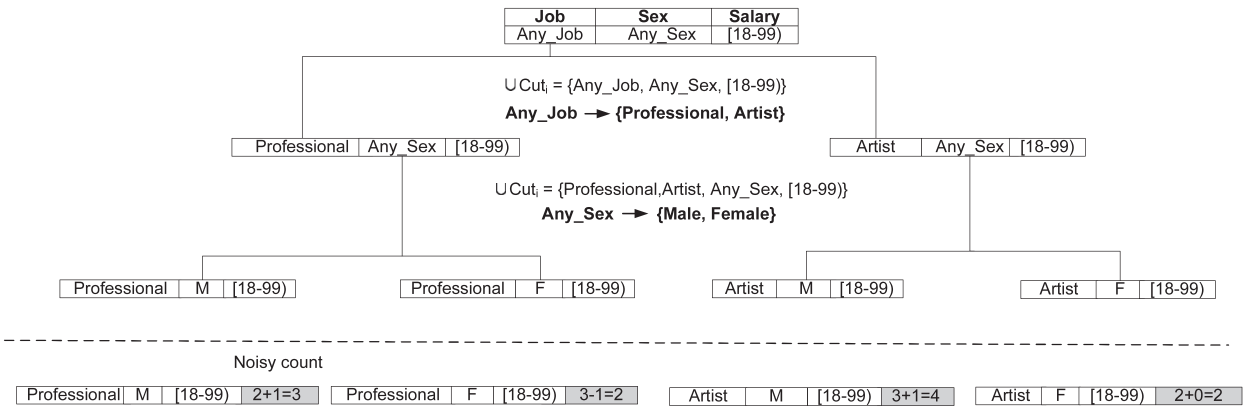

Job | Sex | Salary
Any_Job | Any_Sex | [18-99)
∪Cut_i = {Any_Job, Any_Sex, [18-99)}
Any_Job → {Professional, Artist}
Professional | Any_Sex | [18-99)
Artist | Any_Sex | [18-99)
∪Cut_i = {Professional,Artist, Any_Sex, [18-99)}
Any_Sex → {Male, Female}
Professional | M | [18-99)
Professional | F | [18-99)
Artist | M | [18-99)
Artist | F | [18-99)
Noisy count
Professional | M | [18-99) | 2+1=3
Professional | F | [18-99) | 3-1=2
Artist | M | [18-99) | 3+1=4
Artist | F | [18-99) | 2+0=2

Fig. 2. Generalized data table (D_g). Distributed exponential mechanism is used for specializing the predictor attributes in a top-down manner using half of the privacy budget. Laplace noise is added at leaf nodes to the true count using the second half of the privacy budget to ensure overall ϵ-differentially private output.

(Line 13). If the winner w is not one of P_1's candidates, P_1 waits for instruction from P_2 to specialize w and to update its local copy of $\cup Cut_i$ (Lines 15 and 16). This process is repeated according to the number of the specializations h.

Example 4. Consider the data of Table 1. Initially, D_g contains one root node representing all the records that are generalized to $\langle Any\_Job, Any\_Sex,$[18-99)$\rangle$. $\cup Cut_i$ is represented as $\{Any\_Job, Any\_Sex,$[18-99)$\}$ and includes the initial candidates. To find the winner candidate, both parties run $DistExp$. Suppose the winning candidate w is $Any\_Job \rightarrow \{Professional, Artist\}$. The party P_1 first creates two child nodes under the root node as shown in Fig. 2 and updates $\cup Cut_i$ to {*Professional*, *Artist*, *Any_Sex*, *[18-99)*}. Then, P_1 sends instruction to P_2. On receiving this instruction, P_2 creates the two child nodes under the root node in its copy of D_g and updates the $\cup Cut_i$. Suppose that the next winning candidate is $Any\_Sex \rightarrow \{Male, Female\}$. Similarly, the two parties cooperate to create further specialized partitions resulting in the generalized table in Fig. 2. We do not show the class attribute in Fig. 2.

The split value of a categorical attribute is determined according to the taxonomy tree of the attribute. Since the taxonomy tree is fixed, splitting the records according to the taxonomy tree does not violate differential privacy. For numerical attributes, a split value cannot be directly chosen from the attribute values that appear in the data table D because the probability of selecting the same split value from a different data table D' not containing this value is 0. Therefore, Algorithm 3 uses the exponential mechanism (same as [38]) to determine the split value for each numerical candidate $v^n \in \cup Cut_i$ (Lines 4 and 12).

Computing the count. For each leaf node in the resulted D_g from the previous step, parties need to compute the true count C before adding noise. Using the Secure Scalar Product Protocol [26], [57] (Line 20), parties securely compute the product of the binary vectors Z_1 and Z_2 provided by P_1 and P_2, respectively, to produce the shares C_1 and C_2 of the true count C such that $C = C_1 + C_2$. For each leaf node, the first party P_1 (similarly P_2) computes the binary vector Z_1 such that $|Z_1| = |D_1| = |D_2|$ and $Z_1[i] = 1$ if $D_1[i]$ matches the generalized value of the leaf node; otherwise, $Z_1[i] = 0$.

Example 5 (Continued from Example 4). Consider the bottom most left leaf in Fig. 2, where the count of all male professionals whose salaries in the range *[18-99)* is needed. P_1 generates the binary vector $Z_1 = [0, 0, 0, 0, 1, 1, 1, 0, 1, 1]$, whereas P_2 generates the binary vector $Z_2 = [1, 1, 1, 0, 0, 0, 1, 0, 1, 0]$, as detailed in Table 4. In the secure scalar product protocol, the goal is to securely compute the scalar product $Z_1 * Z_2$ such that

$$\begin{aligned} Z_1 * Z_2 &= \sum_{i=1}^{10} (Z_1[i] \times Z_2[i]) \\ &= 0 + 0 + 0 + 0 + 0 + 0 + 1 + 0 + 1 + 0 = 2. \end{aligned}$$

At the end of the protocol, the two parties have random shares of the result $Z_1 * Z_2$, which is equal to 2.

Computing the noisy count. To compute the overall noisy count, the first party P_1 generates two gaussian random variables $Y_i \sim \mathcal{N}(0, \sqrt{1/\epsilon})$ for $i \in \{1, 2\}$ that are distributed normally with mean 0 and variance $\sqrt{1/\epsilon}$ (Line 21). To clarify why we choose these values for the mean and the variance, we state the following Lemma:

Lemma 6.1 [46]. *Let $Y_i \sim \mathcal{N}(0, \lambda)$ for $i \in \{1, 2, 3, 4\}$ be four Gaussian random variables. Then, the random variable $Lap(2\lambda^2)$ is equal to $Y_1^2 + Y_2^2 - Y_3^2 - Y_4^2$.*

To produce a random variable sampled from a Laplace distribution with parameter $2/\epsilon$, that is, $\mathtt{Lap}(2/\epsilon)$, we need to choose the variance of the Gaussian distribution equal to $\sqrt{1/\epsilon}$. This can be easily verified from Lemma 6.1 by

TABLE 4
Binary Vectors

Shared	Party *A*		Party *B*		
ID	Job	$Z_1[i]$	Sex	Salary	$Z_2[i]$
1	Artist	0	Male	[18-99)K	1
2	Artist	0	Male	[18-99)K	1
3	Artist	0	Male	[18-99)K	1
4	Artist	0	Female	[18-99)K	0
5	Professional	1	Female	[18-99)K	0
6	Professional	1	Female	[18-99)K	0
7	Professional	1	Male	[18-99)K	1
8	Artist	0	Female	[18-99)K	0
9	Professional	1	Male	[18-99)K	1
10	Professional	1	Female	[18-99)K	0

substituting $\lambda = \sqrt{1/\epsilon}$. This gives $2\lambda^2 = 2/\epsilon$. Thus, P_1 computes $C_1 + Y_1^2 - Y_3^2$ to produce its noisy count share X_1, whereas P_2 computes $C_2 + Y_2^2 - Y_4^2$ to produce its noisy count share X_2 (Line 22). The two parties exchange these values to compute the noisy count $C + Lap(2/\epsilon) = X_1 + X_2 = C_1 + Y_1^2 - Y_3^2 + C_2 + Y_2^2 - Y_4^2$ (Line 23).

This simple and efficient method makes the final noisy count $(X_1 + X_2)$ differentially private. However, the exchanged noisy count shares X_1 and X_2 between the parties are not differentially private. The addition of two random Gaussian variables provides some privacy protection against the other party; however, these noisy count shares cannot be differentially private without Laplace noise. We can use instead the proposed technique of Dwork et al. [13] to achieve a flow of information between the parties that is differentially private. Their proposed method uses a simple circuit and takes around 5 seconds for two parties to generate a Laplace noise securely in a collaborative fashion [43]. Thus, a data owner may want to tradeoff privacy for efficiency for noise addition.

6.3 Analysis

We next discuss the correctness, security, and efficiency of Algorithm 3.

Proposition 6.1 (Correctness). *Assuming both parties are semihonest, Algorithm 3 releases ϵ-differentially private data when two parties hold different attributes for the same set of individuals.*

Proof. Algorithm 3 performs exactly the same sequence of operations as in the single-party algorithm DiffGen but in a distributed setting. DiffGen is ϵ-differentially private [38]. Therefore, we prove the correctness of Algorithm 3 by just proving the steps that are different from DiffGen:

- *Candidate selection*. Algorithm 3 selects a candidate for specialization. This step correctly uses the exponential mechanism as stated in Proposition 5.1; therefore, the candidate selection step guarantees ϵ'-differential privacy.
- *Updating the tree D_g and $\cup Cut_i$*. Each party has its own copy of D_g and $\cup Cut_i$. Each party updates these items exactly like DiffGen either by using the local information or by using the instruction provided by the other party.
- *Computing the noisy count*. Algorithm 3 also outputs the noisy count of each leaf node (Line 25), where the noise is equal to $Lap(2/\epsilon)$. Thus, it guarantees $\epsilon/2$-differential privacy.

In summary, Algorithm 3 uses half of the privacy budget to generalize the data (Lines 6-18), where each individual operation is ϵ'-differential privacy; it uses the remaining half of the privacy budget to ensure overall ϵ-differential privacy. □

Proposition 6.2 (Security). *Algorithm 3 is secure under the semihonest adversary model.*

Proof. The security of Algorithm 3 depends on the steps where the two parties exchange information, and it is conducted as follows:

- *Line 7*. The algorithm DistExp is proven to be secure in Section 5.
- *Lines 11 and 15*. The party that owns the winner candidate instructs the other party to specialize w and update its local copy of $\cup Cut_i$. The nature of the top-down approach implies that D_g is more general than the final answer and, therefore, does not leak any additional information.
- *Line 20*. The secure scalar product protocol is proven to be secure [57].
- *Line 23*. The two parties exchange the noisy count shares to compute the noisy count. According to Definition 4.1, the exchange process is secure in a semihonest environment if the additional leakage due to the Gaussian noise is incorporated as a part of the final output. The alternative circuit-based approach is proven to be secure [13].

Therefore, due to composition theorem [21], Algorithm 3 is secure. □

Proposition 6.3 (Complexity). *The encryption and the communication costs of Algorithm 3 are bounded by $O(hnlogC) + 2^h|D|)$ and $O(hnKlogC + 2^he|D|)$, respectively.*

Proof. Most of the encryptions and the communications occur in Line 7 and Line 20 of Algorithm 3. In Line 7, both parties execute distributed exponential mechanism to select a winner candidate. This occurs in a total of h times. Then, according to Proposition 5.3, the number of encryptions and the communication complexity of Line 7 are $O(hnlogC)$ and $O(hnKlogC)$, respectively. In Line 20, parties run SSPP to compute the count of each leaf node. The total number of leaf nodes is 2^h. The encryption and the communication costs of SSPP are $O(|D|)$ and $O(e|D|)$, where e is the bit length of an encrypted item [57]. Therefore, the costs of this step are $O(2^h|D|)$ and $O(2^he|D|)$ for encryption and communication, respectively. Thus, the total costs of the encryption and the communication of Algorithm 3 are bounded by $O(hnlogC + 2^h|D|)$ and $O(hnKlogC + 2^he|D|)$, respectively. □

7 Performance Analysis

In this section, we evaluate the scaling impact on the data utility in terms of classification accuracy. We then compare DistDiffGen with DiffGen [38] and with the distributed algorithm for k-anonymity [39], which we, henceforth, refer to as DAKA. The algorithm DAKA integrates and publishes distributed data with k-anonymity privacy guarantee for classification analysis. Finally, we estimate the computation and the communication costs of DistDiffGen. We employ the publicly available data set *Adult* [15], [18], a real-life census data set that has been used for testing many anonymization algorithms [3], [18], [25], [26], [34], [52], [40]. It has 45,222 census records with six numerical attributes, eight categorical attributes, and a binary class column representing two income levels, ≤50K or >50K. All experiments are conducted on an Intel Core $i7$ 2.7-GHz PC with 12-GB RAM.

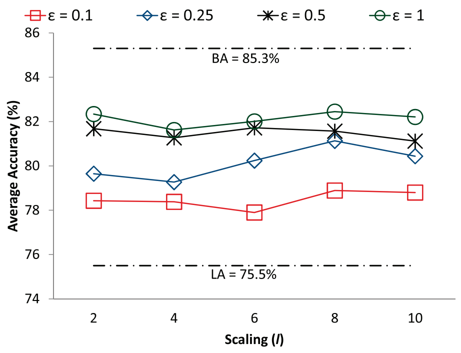


Fig. 3. Classification accuracy for *Adult*, where the number of specialization is $h = 10$. The bottom and top lines stand for Lower-bound Accuracy and Baseline Accuracy, respectively. For LA, the values of the predictor attributes of all the records are generalized to the topmost value in the taxonomy tree. For BA, we use the raw data set without any generalization.

7.1 Experiments

To evaluate the impact on classification quality, we divide the data into training and testing sets. First, we apply our algorithm to anonymize the training set and to determine the $\cup Cut_i$. Then, the same $\cup Cut_i$ is applied to the testing set to produce a generalized testing set. Next, we build a classifier on the anonymized training set and measure the classification accuracy on the generalized records of the testing set. We notice that different partitioning of the attributes among the parties does not have any impact on the classification accuracy, as the winner candidate is chosen from all the candidates of both parties. To compute the score of each candidate $v \in \cup Cut_i$, we adopt the Max utility function [38]:

$$\texttt{Max}(D, v) = \sum_{c \in child(v)} \left(\max_{cls} \left(|D_c^{cls}| \right) \right), \tag{4}$$

where $|D_c^{cls}|$ denotes the number of records in D having generalized value c and the class value cls. Thus, $\texttt{Max}(D, v)$ is the summation of the highest class frequencies over all child values. The sensitivity Δu of the Max function is 1 because the value $\texttt{Max}(D, v)$ can vary at most by 1 due to a record change.

For classification models, we use the well-known C4.5 classifier [45]. To better visualize the cost and the benefits of our approach, we provide additional measures:

1. Baseline accuracy (BA) is the classification accuracy measured on the raw data without anonymization;
2. BA-CA represents the cost in terms of classification quality for achieving a given ϵ-differential privacy requirement;
3. Lower-bound accuracy (LA) is the accuracy on the raw data with all attributes (except for the *Class* attribute) removed—that is, all the predictor attributes for all the records are generalized to the topmost value; and
4. CA-LA represents the benefit of our method over the naive nondisclosure approach.

Fig. 3 depicts the classification accuracy CA for the utility function Max, where the privacy budget $\epsilon \in \{0.1, 0.25, 0.5, 1\}$ and the number of considered digits after the decimal point

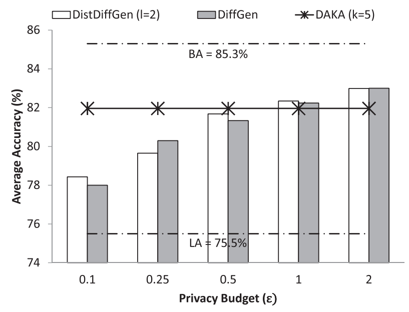


Fig. 4. Comparison of DistDiffGen with DiffGen and DAKA. For DistDiffGen with DiffGen, we vary the privacy budget between $0.1 \leq \epsilon \leq 2$. For DAKA, we fix the anonymity threshold $k = 5$.

$2 \leq l \leq 10$ (i.e., scaling as described in Section 5). The BA and LA are 85.3 and 75.5 percent, respectively, as shown in the figure by the dotted lines. We use 2/3 of the records (i.e., 30,162) to build the classifier and measure the accuracy on the remaining 1/3 of the records (i.e., 15,060). For each experiment, we execute 10 runs and average the results over the runs. The number of specializations h is 10 for all the experiments. For $\epsilon = 1$ and $l = 10$, BA-CA is around 3 percent, whereas CA-LA is 6.7 percent. For $\epsilon = 0.5$, BA-CA spans from 3.58 to 4.18 percent, whereas CA-LA spans from 5.62 to 6.22 percent. However, as ϵ decreases to 0.1, CA quickly decreases to about 78.9 percent (highest point), the cost increases to about 6.4 percent, and the benefit decreases to about 3.4 percent.

We observe two general trends from the experiments. First, the privacy budget has a direct impact on the classification accuracy. A higher budget results in better accuracy because it ensures better attribute partitioning, and it lowers the magnitude of noise that is added to the count of each equivalence group. This observation also holds for DiffGen. Second, the classification accuracy is insensitive to the scaling (the number of the considered digits after the decimal points) for the Max function. This is because the value of $\exp(\frac{\epsilon'}{2\Delta \mathrm{u}} \mathrm{u}(D, v_n))$ is large due to the score of the Max function, which is usually a large integer. Therefore, scaling has hardly any impact on the data utility for classification analysis.

Fig. 4 shows the classification accuracy CA of DistDiffGen, DiffGen, and DAKA. For DiffGen, we use the utility function Max and fix the number of specializations $h = 10$. For DAKA, we fix the anonymity threshold $k = 5$. The accuracy of DistDiffGen is clearly comparable to the one of DiffGen for privacy budget $0.1 \leq \epsilon \leq 2$. The difference of the classification accuracy is due to the randomness introduced by both the exponential and the Laplace mechanisms. The experimental result also shows that DistDiffGen performs better than DAKA for $\epsilon \geq 1$. For a higher anonymity threshold k, the accuracy of DAKA will be lower. This is also expected as DiffGen performed better than TDS [18], a single-party algorithm adopting the k-anonymity privacy model.

In summary, the experimental results demonstrate that the proposed two-party algorithm has properties similar to the single-party algorithm, and the impact of scaling is insignificant.

7.2 Cost Estimate

Most of the computation and the communication take place during the execution of DistExp (Line 7) and SSPP (Line 20). The runtime of the other steps is less than 30 seconds for *Adult* data set. Hence, we only elaborate the runtime of DistExp and SSPP.

7.2.1 Distributed Exponential Mechanism

As discussed in Section 5, the computation and the communication complexity of the distributed exponential mechanism are dominated by the cost of the COMPARISON circuit. In the following, we provide an estimate for the computation and the communication costs of evaluating the COMPARISON circuit. Here, we assume that P_1 encodes and P_2 evaluates the encrypted circuit. The roles of P_1 and P_2 can be swapped.

Computation. For each input bit, P_2 needs to execute a 1-out-of-2 oblivious transfer protocol to get the corresponding encryption key. This is the major computational overhead of the distributed exponential mechanism. The computation cost of an oblivious transfer protocol is roughly equal to the cost of a modular exponentiation, denoted by C_m. Therefore, the computation overhead is equal to the number of input bits of P_2 times C_m. Each input of the circuit is bounded by $\lceil \log_2 C \rceil$ bits, where $C = \sum_{i=1}^{n} \lfloor \mathtt{exp}(\frac{\epsilon'}{2\Delta \mathrm{u}} u(D, v_i)) \times 10^l \rfloor$:

$$\begin{aligned}
\lceil \log_2 C \rceil &= \left\lceil \log_2 \left(\sum_{i=1}^{n} \left\lfloor \mathtt{exp}\left(\frac{\epsilon'}{2\Delta \mathrm{u}} u(D, v_i) \right) \times 10^l \right\rfloor \right) \right\rceil \\
&\leq \left\lceil \sum_{i=1}^{n} \log_2 \left(\left\lfloor \mathtt{exp}\left(\frac{\epsilon'}{2\Delta \mathrm{u}} u(D, v_i) \right) \times 10^l \right\rfloor \right) \right\rceil \\
&\leq \left\lceil \sum_{i=1}^{n} \frac{\frac{\epsilon'}{2\Delta \mathrm{u}} u(D, v_i)}{\ln 2} + \log_2 10^l \right\rceil \\
&= \left\lceil \frac{\epsilon'}{2\Delta \mathrm{u} \ln 2} \sum_{i=1}^{n} u(D, v_i) + (3.3219 \times l) \right\rceil .
\end{aligned}$$

Here, $\Delta \mathrm{u} = 1$, $\epsilon' = \frac{1}{2(6+2\times 10)} = 0.02$, $\sum_{i=1}^{n} u(D, v_i)$ is bounded by the number of the records $|D| = 30{,}162$ for the Max function, and $l = 10$ suffices the desired accuracy. Hence, we have $\lceil \log_2 C \rceil = 469$ bits. The inputs of P_2 are R_2 and L_2 which are 469-bit numbers. As mentioned in Section 6.3, there are at most $h \times n$ invocations of each circuit. Here, n is the total number of candidates, which is 24 at most for the *Adult* data set. Hence, the total computational cost is $h \times n \times 2\lceil \log_2 C \rceil \times C_m = 10 \times 24 \times 2 \times 469 \times 0.02s \approx 75$ minutes, assuming the cost of C_m is 0.02 second for 1,024-bit numbers on a Pentium III processor [42].

Communication. P_1 needs to send a table of size $4K$ for each gate of the COMPARISON circuit, where we assume the key size K is 128 bits. This is the major communication overhead of the distributed exponential mechanism. In Algorithm 2, we describe the COMPARISON circuit that includes two add and one compare circuits. However, two additions and one compare operation can be realized into one circuit. For example, the first two $\lceil \log_2 C \rceil$-bit numbers can be added using $2 \times \lceil \log_2 C \rceil$ binary gates. Thus, we need $4 \times \lceil \log_2 C \rceil$ gates to add R_1, R_2, and L_1, L_2. After the additions, we can compare R and L, which are $\lceil \log_2 C \rceil$-bit numbers, using $5 \times \lceil \log_2 C \rceil - 3$ binary gates [1]. Thus, the total number of gates needed to implement the COMPARISON circuit is $T_g = 9\lceil \log_2 C \rceil - 3 = 4{,}218$. Therefore, the communication cost of sending the tables is $h \times n \times 4K \times T_g \approx 5.18 \times 10^8$ bits, taking approximately 5.5 minutes using a T1 line with 1.544 Mbits/second bandwidth.

Remark. Our estimation ignores the computational cost of evaluating the circuit and the communication cost of the oblivious transfer protocol. The evaluation of the circuit involves decrypting a constant number of ciphertexts (symmetric encryption) for every gate, which is very efficient compared to the oblivious transfer (modular exponentiations) because the number of gates of the circuit is linear to the number of input bits. Also, the communication cost of the oblivious transfer protocol is negligible compared to the cost of sending the tables.

7.2.2 Secure Scalar Product Protocol

We adopt the Secure Scalar Product Protocol of [26] and use its reported running time to estimate the cost of this step for our algorithm. The primary cost of SSPP depends on the number of homomorphic encryptions that is equal to $|D|$, the size of the data set. As reported in [26], the estimated cost of the homomorphic encryptions is 19.5 s on average when $|D| = 30162$ (the size of our data set) on Intel Xeon 3-GHz processor. The computation cost of Line 20 is $2^h \times 19.5 \text{ s} = 2^{10} \times 19.5 \text{ s} \approx 5.5$ hours. Notice that Line 20 is easily parallelizable because each leaf node pair is independent. Assuming we have 10 processors, this cost can be reduced to 33 minutes. Finally, the communication cost is $2^h \times e \times |D| = 2^{10} \times 1{,}024 \times 30{,}162 = 3.16 \times 10^{10}$ bits, assuming $e = 1{,}024$ bits. Thus, the communication overhead of Line 20 is around 5.6 hours using a T1 line. This time can be reduced to 15 minutes by using a T3 line with 35 Mbits/second bandwidth.

8 Discussion

Is differential privacy good enough? What changes are required if there are more than two parties? How reasonable is it to assume that the parties are semihonest? In this section, we provide answers to these questions.

Differential Privacy. Differential privacy is a strong privacy definition. However, Kifer and Machanavajjhala [29] have shown that if the records are not independent or an adversary has access to aggregate level background knowledge about the data, then privacy attack is possible. In our application scenario, each record is independent of the other records and we assume that no deterministic statistics of the raw database have ever been released. Hence, differential privacy is appropriate for our problem.

More than two parties. The proposed algorithm is only applicable for the two-party scenario because the distributed exponential algorithm and the other primitives (e.g., random value protocol, secure scalar product protocol) are limited to a two-party scenario. The proposed algorithm can be extended for more than two parties by modifying all the subprotocols while keeping the general top-down structure of the algorithm.

Semihonest Adversary Model. This is the common security definition used in the SMC literature [26]; it is realistic in our problem scenario because different organizations are collaborating to securely share their data for mutual benefits. Hence, it is reasonable to assume that parties will not deviate from the defined protocol. However, they may be curious to learn additional information from the messages they received during the protocol execution. To extend the algorithm for malicious parties, all subprotocols should be extended and must be secure under the malicious adversary model.

9 Conclusion

In this paper, we have presented the first two-party differentially private data release algorithm for vertically partitioned data. We have shown that the proposed algorithm is differentially private and secure under the security definition of the semihonest adversary model. Moreover, we have experimentally evaluated the data utility for classification analysis. The proposed algorithm can effectively retain essential information for classification analysis. It provides similar data utility compared to the recently proposed single-party algorithm [38] and better data utility than the distributed k-anonymity algorithm for classification analysis [39].

Acknowledgments

The research was supported in part by the Discovery Grants (356065-2013) and Canada Graduate Scholarships from the Natural Sciences and Engineering Research Council of Canada.

References

[1] R. Agrawal, A. Evfimievski, and R. Srikant, "Information Sharing Across Private Databases," *Proc. ACM Int'l Conf. Management of Data*, 2003.

[2] B. Barak, K. Chaudhuri, C. Dwork, S. Kale, F. McSherry, and K. Talwar, "Privacy Accuracy, and Consistency Too: A Holistic Solution to Contingency Table Release," *Proc. ACM Symp. Principles of Database Systems (PODS '07)*, 2007.

[3] R.J. Bayardo and R. Agrawal, "Data Privacy through Optimal k-Anonymization," *Proc. IEEE Int'l Conf. Data Eng. (ICDE '05)*, 2005.

[4] R. Bhaskar, S. Laxman, A. Smith, and A. Thakurta, "Discovering Frequent Patterns in Sensitive Data," *Proc. ACM Int'l Conf. Knowledge Discovery and Data Mining (SIGKDD '10)*, 2010.

[5] A. Blum, K. Ligett, and A. Roth, "A Learning Theory Approach to Non-Interactive Database Privacy," *Proc. ACM Symp. Theory of Computing (STOC '08)*, 2008.

[6] J. Brickell and V. Shmatikov, "Privacy-Preserving Classifier Learning," *Proc. Int'l Conf. Financial Cryptography and Data Security*, 2009.

[7] P. Bunn and R. Ostrovsky, "Secure Two-Party K-Means Clustering," *Proc. ACM Conf. Computer and Comm. Security (CCS '07)*, 2007.

[8] K. Chaudhuri, C. Monteleoni, and A. Sarwate, "Differentially Private Empirical Risk Minimization," *J. Machine Learning Research*, vol. 12, pp. 1069-1109, July 2011.

[9] K. Chaudhuri, A.D. Sarwate, and K. Sinha, "Near-Optimal Differentially Private Principal Components," *Proc. Conf. Neural Information Processing Systems*, 2012.

[10] C. Clifton, M. Kantarcioglu, J. Vaidya, X. Lin, and M.Y. Zhu, "Tools for Privacy Preserving Distributed Data Mining," *ACM SIGKDD Explorations Newsletter*, vol. 4, no. 2, pp. 28-34, Dec. 2002.

[11] I. Dinur and K. Nissim, "Revealing Information while Preserving Privacy," *Proc. ACM Symp. Principles of Database Systems (PODS '03)*, 2003.

[12] C. Dwork, "A Firm Foundation for Private Data Analysis," *Comm. ACM*, vol. 54, no. 1, pp. 86-95, 2011.

[13] C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor, "Our Data Ourselves: Privacy via Distributed Noise Generation," *Proc. 25th Ann. Int'l Conf. Theory and Applications of Cryptographic Techniques (EUROCRYPT '06)*, 2006.

[14] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating Noise to Sensitivity in Private Data Analysis," *Proc. Theory of Cryptography Conf. (TCC '06)*, 2006.

[15] A. Frank and A. Asuncion, UCI Machine Learning Repository, http://mlearn.ics.uci.edu/MLRepository.html, 2010.

[16] A. Friedman and A. Schuster, "Data Mining with Differential Privacy," *Proc. ACM Int'l Conf. Knowledge Discovery and Data Mining (SIGKDD '10)*, 2010.

[17] B.C.M. Fung, K. Wang, R. Chen, and P.S. Yu, "Privacy-Preserving Data Publishing: A Survey of Recent Developments," *ACM Computing Surveys*, vol. 42, no. 4, pp. 1-53, June 2010.

[18] B.C.M. Fung, K. Wang, and P.S. Yu, "Anonymizing Classification Data for Privacy Preservation," *IEEE Trans. Knowledge and Data Eng.*, vol. 19, no. 5, pp. 711-725, May 2007.

[19] S.R. Ganta, S. Kasiviswanathan, and A. Smith, "Composition Attacks and Auxiliary Information in Data Privacy," *Proc. ACM Int'l Conf. Knowledge Discovery and Data Mining (SIGKDD '08)*, 2008.

[20] O. Goldreich, "A Note on Computational Indistinguishability," *Information Processing Letter*, vol. 34, no. 6, pp. 277-281, 1990.

[21] O. Goldreich, *Foundations of Cryptography*, vol. 2, Cambridge Univ. Press, 2001.

[22] O. Goldreich, S. Micali, and A. Wigderson, "How to Play Any Mental Game—A Completeness Theorem for Protocols with Honest Majority," *Proc. ACM Symp. Theory of Computing (STOC '87)*, 1987.

[23] M. Hay, V. Rastogi, G. Miklau, and D. Suciu, "Boosting the Accuracy of Differentially Private Histograms through Consistency," *Proc. Int'l Conf. Very Large Data Bases (VLDB '10)*, 2010.

[24] A. Inan, M. Kantarcioglu, G. Ghinita, and E. Bertino, "Private Record Matching Using Differential Privacy," *Proc. Int'l Conf. Extending Database Technology (EDBT '10)*, 2010.

[25] V.S. Iyengar, "Transforming Data to Satisfy Privacy Constraints," *Proc. ACM Int'l Conf. Knowledge Discovery and Data Mining (SIGKDD '02)*, 2002.

[26] W. Jiang and C. Clifton, "A Secure Distributed Framework for Achieving k-Anonymity," *Very Large Data Bases J.*, vol. 15, no. 4, pp. 316-333, Nov. 2006.

[27] P. Jurczyk and L. Xiong, "Distributed Anonymization: Achieving Privacy for Both Data Subjects and Data Providers," *Proc. Ann. IFIP WG 11.3 Working Conf. Data and Applications Security (DBSec '09)*, 2009.

[28] D. Kifer, "Attacks on Privacy and De Finetti's Theorem," *Proc. ACM Conf. Management of Data (SIGMOD '09)*, 2009.

[29] D. Kifer and A. Machanavajjhala, "No Free Lunch in Data Privacy," *Proc. ACM Conf. Management of Data (SIGMOD '11)*, 2011.

[30] K. LeFevre, D.J. DeWitt, and R. Ramakrishnan, "Mondrian Multidimensional K-Anonymity," *Proc. IEEE Int'l Conf. Data Eng. (ICDE '06)*, 2006.

[31] K. LeFevre, D.J. DeWitt, and R. Ramakrishnan, "Workload-Aware Anonymization Techniques for Large-Scale Data Sets," *ACM Trans. Database Systems*, vol. 33, article 17, 2008.

[32] N. Li, T. Li, and S. Venkatasubramanian, "t-Closeness: Privacy Beyond k-Anonymity and ℓ-Diversity," *Proc. IEEE Int'l Conf. Data Eng. (ICDE '07)*, 2007.

[33] Y. Lindell and B. Pinkas, "Privacy Preserving Data Mining," *J. Cryptology*, vol. 15, no. 3, pp. 177-206, 2002.

[34] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam, "ℓ-Diversity: Privacy Beyond k-Anonymity," *ACM Trans. Knowledge Discovery from Data*, vol. 1, article 3, 2007.

[35] D. Martin, D. Kifer, A. Machanavajjhala, J. Gehrke, and J. Halpern, "Worst-Case Background Knowledge in Privacy-Preserving Data Publishing," *Proc. IEEE Int'l Conf. Data Eng. (ICDE '07)*, 2007.

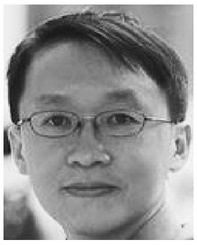
[36] A. McGregor, I. Mironov, T. Pitassi, O. Reingold, K. Talwar, and S. Vadhan, "The Limits of Two-Party Differential Privacy," *Proc. IEEE Symp. Foundations of Computer Science (FOCS '10)*, 2010.

[37] F. McSherry and K. Talwar, "Mechanism Design via Differential Privacy," *Proc. IEEE Symp. Foundations of Computer Science (FOCS '07)*, 2007.

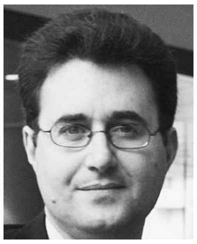
[38] N. Mohammed, R. Chen, B.C.M. Fung, and P.S. Yu, "Differentially Private Data Release for Data Mining," *Proc. ACM Int'l Conf. Knowledge Discovery and Data Mining (SIGKDD '11),* 2011.
[39] N. Mohammed, B.C.M. Fung, and M. Debbabi, "Anonymity Meets Game Theory: Secure Data Integration with Malicious Participants," *Very Large Data Bases J.,* vol. 20, no. 4, pp. 567-588, Aug. 2011.
[40] N. Mohammed, B.C.M. Fung, P.C.K. Hung, and C. Lee, "Anonymizing Healthcare Data: A Case Study on the Blood Transfusion Service," *Proc. ACM Int'l Conf. Knowledge Discovery and Data Mining (SIGKDD '09),* 2009.
[41] N. Mohammed, B.C.M. Fung, P.C.K. Hung, and C. Lee, "Centralized and Distributed Anonymization for High-Dimensional Healthcare Data," *ACM Trans. Knowledge Discovery from Data,* vol. 4, no. 4, pp. 18:1-18:33, Oct. 2010.
[42] M. Naor and B. Pinkas, "Efficient Oblivious Transfer Protocol," *Proc. 12th Ann. ACM-SIAM Symp. Discrete Algorithms (SODA '01),* 2001.
[43] A. Narayan and A. Haeberlen, "DJoin: Differentially Private Join Queries over Distributed Databases," *Proc. 10th USENIX Conf. Operating Systems Design and Implementation (OSDI '12),* 2012.
[44] P. Paillier, "Public-Key Cryptosystems Based on Composite Degree Residuosity Classes," *Proc. 17th Int'l Conf. Theory and Application Cryptographic Techniques,* pp. 223-238, 1999.
[45] J.R. Quinlan, *C4.5: Programs for Machine Learning.* Morgan Kaufmann, 1993.
[46] V. Rastogi and S. Nath, "Differentially Private Aggregation of Distributed Time-Series with Transformation and Encryption," *Proc. ACM Int'l Conf. Management of Data (Sigmod '10),* 2010.
[47] A. Roth and T. Roughgarden, "Interactive Privacy via the Median Mechanism," *Proc. ACM Symp. Theory of Computing (STOC '10),* 2010.
[48] P. Samarati, "Protecting Respondents' Identities in Microdata Release," *IEEE Trans. Knowledge and Data Eng.,* vol. 13, no. 6, pp. 1010-1027, Nov. 2001.
[49] L. Sweeney, "k-Anonymity: A Model for Protecting Privacy," *Int'l J. Uncertainty, Fuzziness and Knowledge-Based Systems,* vol. 10, pp. 557-570, 2002.
[50] J. Vaidya and C. Clifton, "Privacy Preserving Association Rule Mining in Vertically Partitioned Data," *Proc. ACM Int'l Conf. Knowledge Discovery and Data Mining (SIGKDD '02),* 2002.
[51] J. Vaidya and C. Clifton, "Privacy-Preserving k-Means Clustering over Vertically Partitioned Data," *Proc. ACM Int'l Conf. Knowledge Discovery and Data Mining (SIGKDD '03),* 2003.
[52] K. Wang, B.C.M. Fung, and P.S. Yu, "Handicapping Attacker's Confidence: An Alternative to k-Anonymization," *Knowledge and Information Systems,* vol. 11, no. 3, pp. 345-368, Apr. 2007.
[53] O. Williams and F. McSherry, "Probabilistic Inference and Differential Privacy," *Proc. Conf. Neural Information Processing Systems (NIPS '10),* 2010.
[54] R.C.W. Wong, A.W.C. Fu, K. Wang, and J. Pei, "Minimality Attack in Privacy Preserving Data Publishing," *Proc. Int'l Conf. Very Large Data Bases,* 2007.
[55] R.C.W. Wong, A.W.C. Fu, K. Wang, Y. Xu, and P.S. Yu, "Can the Utility of Anonymized Data be Used for Privacy Breaches?" *ACM Trans. Knowledge Discovery from Data,* vol. 5, no. 3, article 16, Aug. 2011.
[56] R.C.W. Wong, J. Li, A.W.C. Fu, and K. Wang, "(α, k)-Anonymity: An Enhanced k-Anonymity Model for Privacy Preserving Data Publishing," *Proc. ACM Int'l Conf. Knowledge Discovery and Data Mining (SIGKDD '06),* 2006.
[57] R. Wright and Z. Yang, "Privacy-Preserving Bayesian Network Structure Computation on Distributed Heterogeneous Data," *Proc. ACM Int'l Conf. Knowledge Discovery and Data Mining (SIGKDD '04),* 2004.
[58] X. Xiao, G. Wang, and J. Gehrke, "Differential Privacy via Wavelet Transforms," *Proc. IEEE Int'l Conf. Data Eng.,* 2010.
[59] A.C. Yao, "Protocols for Secure Computations," *Proc. IEEE Symp. Foundations of Computer Science (FOCS '82),* 1982.
[60] L. Zhang, S. Jajodia, and A. Brodsky, "Information Disclosure under Realistic Assumptions: Privacy versus Optimality," *Proc. ACM Conf. Computer and Comm. Security (CCS '07),* 2007.

Noman Mohammed received the MASc degree in information systems security and the PhD degree in computer science, both from Concordia University, Canada. He is a postdoctoral fellow in the School of Computer Science at McGill University. His research interests include private data analysis, secure distributed computing, trustworthy cloud computing, and wireless network security.

Dima Alhadidi received the PhD degree in computer science and software engineering from Concordia University, Canada. She is currently a research associate at Concordia University. Her research interests include computer security, aspect-oriented programming, software engineering, secure distributed computing, and cloud computing.



Benjamin C.M. Fung received the PhD degree in computing science from Simon Fraser University in 2007. He is an associate professor at McGill University, Montreal, Canada, and a research scientist of the National Cyber-Forensics and Training Alliance of Canada. He has more than 60 refereed publications that span across the prestigious research forums of data mining, privacy protection, cyber forensics, web services, and building engineering. He is a senior member of the IEEE.



Mourad Debbabi received the MSc and PhD degrees in computer science from Paris-XI Orsay University, France. He is currently a full professor and the director of the Concordia Institute for Information Systems Engineering, Concordia University, Montreal, Quebec, Canada. He holds the Concordia Research Chair Tier I in Information Systems Security. He is also the president of the National Cyber-Forensics Training Alliance of Canada. He has published more than 170 research papers in journals and conferences on computer security, digital forensics, Java security, cryptographic protocols, malicious code detection, and network security.