Efficient and Secure Outsourcing of Differentially Private Data Publishing with Multiple Evaluators

Jin Li, Heng Ye, Tong Li, Wei Wang, Wenjing Lou, Y. Thomas Hou, Jiqiang Liu, and Rongxing Lu

Abstract—Since big data becomes a main impetus to the next generation of IT industry, data privacy has received considerable attention in recent years. To deal with the privacy challenges, differential privacy has been widely discussed and related private mechanisms are proposed as privacy-enhancing techniques. However, with today's differential privacy techniques, it is difficult to generate a sanitized dataset that can suit every machine learning task. In order to adapt to various tasks and budgets, different kinds of privacy mechanisms have to be implemented, which inevitably incur enormous costs for computation and interaction. To this end, in this paper, we propose two novel schemes for outsourcing differential privacy. The first scheme efficiently achieves outsourcing differential privacy by using our preprocessing method and secure building blocks. To support the queries from multiple evaluators, we give the second scheme that employs a trusted execution environment to aggregately implement privacy mechanisms on multiple queries. During data publishing, our proposed schemes allow providers to go off-line after uploading their datasets, so that they achieve a low communication cost which is one of the critical requirements for a practical system. Finally, we report an experimental evaluation on UCI datasets, which confirms the effectiveness of our schemes.

Index Terms—Differential privacy, cloud computing, outsourcing, encryption

1 Introduction

There is a general consensus that we are currently in the era of big data, with tremendous amounts of information being collected by various organizations and entities. Often, these data providers may wish to contribute their data to studies involving tasks such as statistical analysis, classification, and prediction. Because cloud service providers (CSPs) offer data providers (owners) great flexibility with respect to computation and storage capabilities, CSPs are presently the most popular avenue, through which data providers can share their data. However, the risk of leaking individuals' private information with the straightforward uploading of data providers' data (which may contain sensitive information such as medical or financial records, addresses and telephone numbers, or preferences of various kinds that the individuals may not want exposed) to CSPs is unacceptable. In such a situation, differential privacy (DP) has been considered as a privacy standard not only for achieving an acceptable trade-off between data usability and data privacy. As shown in Fig. 1, there are generally two main frameworks that can be used for achieving DP, i.e., the framework with interaction for release, and the framework with no interaction for publishing. In this paper, we will focus on the latter, i.e., differentially private publishing.

In the previous work [1], we gave a the notion of outsourcing differentially private data publishing. However, there does not exist an efficient method that allows data to be published only once while still preserving the data's utility for multiple evaluation algorithms and applications. On one hand, when providers' data need to be shared for uses involving different tasks, different kinds of privacy mechanisms have to be implemented on released data. That is, datasets should be set appropriately to answer different queries in different privacy budget/function of privacy mechanisms. Consequently, the computation overhead and interaction will be enormous if the number of queries is large. On the other hand, when data providers publish their data, public entities must exist somewhere that can store all of the different kinds of data for applying different privacy mechanisms, which also inevitably requires considerable storage space.

To address these two kinds of challenges, we extend the method of outsourcing differentially private data publishing in this paper. With the advent of cloud computing, we know an increasing number of storage and computing tasks are moving from local resources to CSPs. In our scheme, the implementation of privacy mechanisms is outsourced to a CSP by the data providers. Considering the CSP is not totally trusted, providers wish to protect sensitive information in their datasets by secure techniques (e.g.,

- *J. Li is with the School of Computer Science and Cyber Engineering, Guangzhou University, Guangzhou, China and the Department of Computer Science, Virginia Polytechnic Institute and State University, Blacksburg, VA 24061, USA. Email: jinli71@gmail.com.*
  H. Ye, Wei Wang, and J. Liu are with Beijing Key Laboratory of Security and Privacy in Intelligent Transportation, Beijing Jiaotong University, Beijing, China. Email: heng.ye@bjtu.edu.cn, wangwei1@bjtu.edu.cn, and jqliu@bjtu.edu.cn.
  T. Li is with the School of Computer Science and Cyber Engineering, Guangzhou University, Guangzhou, China and the College of Cyber Science and the College of Computer Science, Nankai University, Tianjin, China. Email: litongziyi@mail.nankai.edu.cn.
  W. Lou is with the Department of Computer Science, Virginia Polytechnic Institute and State University, Blacksburg, VA 24061, USA. Email: wjlou@vt.edu.
  Y. T. Hou is with the Department of Electrical and Computer Engineering, Virginia Polytechnic Institute and State University, Blacksburg, VA 24061, USA. Email: thou@vt.edu.
  R. Lu is with the Faculty of Computer Science, University of New Brunswick, Fredericton, NB, Canada. Email: RLU1@unb.ca.

- *This paper is the extended version of the conference paper [1].*

cryptographic schemes) before outsourcing the datasets. However, to support ciphertext manipulations for the differentially private data release/publishing, some secure manners, such as fully homomorphic encryption (FHE) schemes, will certainly introduce enormous amounts of storages and communication bandwidths. To this end, our differentially private data publishing scheme relies on some secure building blocks which would be much more efficient for implementing privacy mechanisms in an outsourcing way. Moreover, to expand the form of records and the type of queries for multiple evaluators, we proposed an advance scheme based on a trusted execution environment to achieve differential privacy aggregately. In both of our schemes, the data providers are not required to be on-line when their data are requested, which is one of the critical requirements for a practical application system.

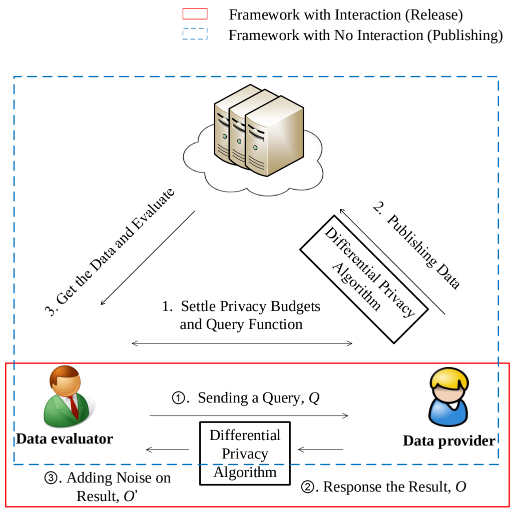

🟥 Framework with Interaction (Release)
Framework with No Interaction (Publishing)
3. Get the Data and Evaluate
2. Publishing Data
Differential Privacy Algorithm
1. Settle Privacy Budgets and Query Function
①. Sending a Query, Q
Data evaluator
Data provider
Differential Privacy Algorithm
③. Adding Noise on Result, O'
②. Response the Result, O

Figure 1: Frameworks for Achieving Differential Privacy

1.1 Related Work

Differential privacy has been accepted as the main privacy paradigm in recent years, because it is based on purely mathematical calculations and provides a means of quantitative assessment. A large body of work on DP has accumulated due to its support for privacy preserving data analysis. There are some works which used the cryptographic methods to solve the privacy preserving for data utilization [2], [3] before the first work of DP by Dwork in 2006 [4], and the Laplace mechanism for adding noise to achieve ϵ-DP was proposed in the seminal paper. Subsequently, McSherry designed the exponent mechanism [5] and identified the sequential and parallel properties of DP [6].

Generally, DP can be achieved via two main frameworks. In the first framework, the data evaluator's queries are responsed under a predetermined privacy budget ϵ; as shown in Fig 1 with red box. In the framework, we can apply the Laplace [7], Privlet [8], Linear Query [9], and Batch Query [10] techniques, among others, to obtain different responses to these queries that satisfy ϵ-DP. However, this framework demands interaction between the data provider and the data evaluator. The second framework, as depicted in Fig. 1 with the blue dashed box, does not require any interaction. The main focus of research related to this framework is on how to design efficient and effective noise-adding algorithms to ensure DP while boosting data utility.

Typical publishing methods include histogram publishing [11]–[14], partition publishing [15]–[18], contingency table publishing [19], [20], and sanitized dataset publishing [21]–[24]. A histogram is an intuitive representation of the distribution of a set of data and can be used as a basis for other statistical queries or linear queries. However, histogram publishing suffers from problems of redundant noise and inconsistency, meaning that different types of noise should be added for different uses. Partition publishing can reduce the amount of noise that must be added. The foundation of partition publishing is the careful design of an index structure to support the partitioning of the data. Using this index, the data provider can assign a privacy budget to each partition for noise addition before publishing. However, determining how to assign a privacy budget is not a trivial task, and the partition index itself may leak some sensitive information; this potential risk is the core problem that remains to be solved for this method of publishing. Often, data can be represented in the form of a contingency table. In fact, instead of publishing the contingency table itself for analysis, data are often published based on the statistical values of the combinations of certain variables, as represented by marginal tables. Directly adding noise to a contingency table introduces too much noise, whereas perturbing the marginal table may cause inconsistency. Qardaji proposed a noise-adding method in which the contingency table is divided into small pieces, called views [25]. This method can reduce the amount of noise introduced, but the questions of how to choose the parameters to be used for division and how to preserve the consistency between the views and the marginal tables remain challenging. The purpose of sanitized dataset publishing is to ensure the protection of data privacy after the processing of the original dataset. The question on how to directly publish a sanitized dataset that satisfies DP while allowing the data evaluator to make any necessary inquiries is quite challenging. This method of dataset publishing demands considerable calculation and thus is inefficient and difficult to realize. Kasiviswanathan and Blum proved that sanitized dataset publishing is possible [26], [27]; however, it requires an enormous number of records.

The multi-party DP scheme was first proposed by Manas in 2010 [28], based on the aggregation of multi-party datasets to train a classifier. Subsequently, many works involving multi-party DP publishing have been reported, including multi-task learning [29], multi-party deep learning [30], classifier training on private and public datasets [31] and high-dimensional data publishing [32]. These works, however, have not considered outsourced computing to relieve the computational burden on data providers. Our initial work [1] can provide frameworks for data evaluators to efficiently analyze databases belonging to a single party, but it has efficiency bottlenecks in the multi-party setting.

Outsourced computing is a technique for securely outsourcing expensive computations to untrusted servers, which allows resource-constrained data providers to outsource their computational workloads to cloud servers with unlimited computational resources. Chaum first proposed the notion of wallets, secure hardware installed on a client's computer to perform expensive computations, in 1992 [33]. To protect data providers' privacy, Chevallier ppresented the first algorithm for the secure delegation of elliptic-curve pairings [34]. In addition, solutions for performing meaningful computations over encrypted data using fully homomorphic encryption have emerged, although they are known to be of poor practicality [35].

Meanwhile, cloud computing using attribute-based encryption appeared, which not only supports ciphertext operation, but also provides fine-grained access control. Some of works that concentrate on privacy preserving in cloud computing have been proposed recentlyp [36], [37]. Nevertheless, in this paper, we choose additive homomorphic encryption as our basic technique to perform outsourcing computing, which offers a perfect balance of efficiency and security.

The software guard extensions (SGX) is a set of extensions to the Intel architecture that aims to provide integrity and confidentiality guarantees for computations performed on an untrusted or remote hardware where all the privileged software (kernel, hypervisor, etc) is potentially malicious. Due to these properties, the SGX is widely applied for secure outsourcing computations [38], including most machine learning algorithms such as, SVM, matrix decomposition, neural network, decision tree, and k-means [39]–[44].

1.2 Contributions

In this paper, we propose two novel outsourced DP data publishing schemes for cloud computing. Our contributions can be summarized as follows.

- Based on our previous work, we design a new preprocessing method and give an efficient outsourced data publishing approach using our secure building blocks instead of fully homomorphic encryption. In such a way, data can be efficiently outsourced to a CSP for secure storage and implementation of differential privacy mechanisms.
- For supporting multiple evaluators, We design an advance outsourced scheme based on SGX to aggregate data providers' datasets. In such a way, the responses of data evaluators' queries can achieve differential privacy in an efficient way.
- In our scheme, the data provider is not required to be involved in subsequent noise computations and related processing.
- The security of the data against the CSP can be guaranteed.

1.3 Organization

The rest of this paper is organized as follows. Some preliminary considerations are discussed in Section 2. In Section 3, the architecture of our scheme and our threat model are introduced. Then, we present the problem statement and describe our schemes in detail in Section 4. In Section 5, we analyze the security of the proposed scheme. The implementation details and the evaluation of the experimental results are presented in Section 6. Finally, we conclude our work in Section 7.

2 Preliminaries

2.1 Differential Privacy

DP was introduced by Dwork et al. as a privacy standard for individual privacy protection in data publishing. It provides a strong privacy guarantee, ensuring that the presence or absence of an individual will not significantly affect the final output of any function.

Definition 1. *(Differential privacy)*
A randomized function $\mathcal{A}$ with a well-defined probability density $\mathcal{P}$ satisfies ϵ-DP if, for any two neighboring datasets D_1 and D_2 that differ by only one record and for any $\mathcal{O} \in range(M)$,

$$\mathcal{P}(\mathcal{A}(D_1) = \mathcal{O}) \leq e^{\epsilon} \cdot \mathcal{P}(\mathcal{A}(D_2) = \mathcal{O}) \tag{1}$$

The sensitivity of a queried function f on the database is an important concept for implementing privacy mechanisms such as Laplace mechanism and the exponential mechanism. It is defined as follows.

Definition 2. *(Global sensitivity)*
Let f be a function that maps a database to a fixed-size vector of real numbers. For all neighboring databases D_l and D_2, the global sensitivity of f is defined as

$$\Delta(f) = \max_{D_1, D_2} \|f(D_1) - f(D_2)\|_1 \tag{2}$$

where $\|\cdot\|_1$ denotes the L_1 norm.

To publish data that satisfy ϵ-DP for a queried function f, the data provider should perturb the output of f by the sensitivity Δf and the privacy budget ϵ. For example, for the Laplace mechanism, let Lap(λ) denote the Laplace probability distribution with mean zero and scale λ. The Laplace mechanism achieves DP by adding Laplace noise to the query result $f(M)$.

2.2 Homomorphic Encryption

We use $[[\cdot]]$ to denote ciphertexts of homomorphic encryption schemes in this paper. We say that an encryption scheme is fully homomorphic if it conforms to the following definition.

Definition 3. *(Fully Homomorphic Encryption)*
Let m_1 and m_2 be two plaintexts, let $\mathcal{A}$ be an encryption algorithm that outputs the corresponding ciphertexts $[[m_1]]$ and $[[m_2]]$, and let $\mathcal{B}$ be an operation (addition/multiplication) performed on the two ciphertexts. For any two ciphertexts, fully homomorphic encryption (FHE) has the following property:

$$\mathcal{B}([[m_1]], [[m_2]]) = [[\mathcal{B}(m_1, m_2)]] \tag{3}$$

Similarly, the encryption scheme is additive homomorphic since it conforms to the following definition.

Definition 4. *(Additive homomorphic encryption)*
Let m_1 and m_2 be two plaintexts, let $\mathcal{A}$ be an encryption algorithm that outputs the corresponding ciphertexts $[[m_1]]$ and $[[m_2]]$, and let $\mathcal{B}$ be an operation performed on the two ciphertexts.

For any two ciphertexts, additive homomorphic encryption (AHE) has the following property:

$$\mathcal{B}([[m_1]],[[m_2]]) = \mathcal{B}(\mathcal{A}(m_1),\mathcal{A}(m_2)) = [[m_1+m_2]] \quad (4)$$

2.3 Intel SGX

Intel SGX is a trusted execution environment for executing secure code in Intel processors. Programmers need to partition the code into trusted and untrusted components. The trusted code is encrypted and integrity protected, but the untrusted code is observable by the operating system. During the program execution, the untrusted component creates a secure component inside the processor called enclave and loads trusted code into it. After creating the enclave, users can verify that intended code is loaded and securely provision the code with Intel's secret keys, which is called attestation. In addition, trusted and untrusted components communicate between each other using programmer defined entry points. Entry points defined in trusted code is called ECalls, which can be called by untrusted part once enclave is loaded. Similarly, entry points defined in untrusted code is called OCalls, which can be called by the trusted part.

In another word, the SGX enclaves aim at isolating execution of a program from all other processes on a untrusted host, but the enclave memory is fully encrypted and authenticated. This property provide remote attestations that a remote party can verify using Intel's public key. It is the security guarantees which SGX-based schemes rely on.

3 Architecture

In this section, we formalize our system model, and identify the threat model and security requirements.

3.1 System Model

In our system model, three types of entities are involved in our basic scheme, namely, the data providers, the cloud service provider (CSP), and the data evaluator. The data providers possess data and would like to share those data for purposes such as machine learning or data analysis. The CSP provides the cloud storage service for the data providers. The data evaluator makes queries for data analysis and finally obtains the differentially private results. Each data evaluator may acquire different part of data for different usage.

3.2 Threat Model and Security Requirements

In this work, both the data evaluator and the CSP are assumed to be *honest-but-curious*. The data evaluator needs to protect his trained model against the CSP. Moreover, the data evaluator will follow a specified protocol for building a correct model without obtaining incorrect results. The privacy of the data providers' data needs to be protected against both the CSP and the data evaluator.

For the data owner, the security means that its privacy should be protected against the other entities even if they are curious about the underlying original data. Of course, the CSP and DP are not allowed to collude with each other in our security model.

Suppose a data provider want to publish his data, our goal is to process his data to meet the following security requirements.

Data privacy: The query results should satisfy ϵ-differential privacy.

Data utility: The query results should preserve as much information as possible for classification analysis.

4 Outsourced Private Data Publishing Schemes

In this section, we give our outsourced private data publishing schemes. In more details, we apply some popular privacy mechanisms, such as Laplace mechanism and the exponential mechanism, as concrete constructions.

4.1 A Straightforward Scheme

Before the presentation of our schemes, there is a straightforward scheme used in outsourced differential privacy, where a FHE is used. The typical process of differentially private data publishing can be summarized as follows:

- **Setup**. The data evaluator and data provider together establish the keys they used (pk, sk).
- **Uploading**. The data provider uploads the encrypted dataset $[M]_{pk}$ to the CSP using fully homomorphic encryption.
- **Data processing**. CSP get the query f from data evaluator and calculate the result in encrypted form $f([[M]]_{pk}) = [[f(M)]]_{pk}$. After the calculation of global sensitity Δf, CSP generates the noise $\eta \sim Lap(\Delta f/\epsilon)$ and add it on the result $[[f(M)+\eta]]_{pk}$.
- **Analysis**. The data evaluator obtains the encrypted result with noise, decrypts it and analyzes it.

Generally speaking, the storage security of this scheme is mainly based on FHE. The data privacy got a full rely on the ϵ-DP. However, for different queries or different privacy budgets, those steps mentioned above must be repeated, which is inefficient, waste a lot of computing power.

4.2 An Advanced Scheme with AHE-based Building Blocks

As we know, current non-interactive approaches usually publish contingency tables or marginals of the raw data. Most of these approaches will draw a frequency matrix of the raw data over the database domain. After that, noise is added to each count to satisfy the privacy requirement before the publishing of noisy frequency matrix. In order to support more machine learning tasks and more privacy budgets, lots of data pre-processing work should be done by data providers. In other words, data provider will consume a lot of his computing power to construct different layers of his data, which is a limit of our scheme. Also, even the CSP will choose the suitable layer of data (count) to add noise, there are few queries (range queries, count queries, etc...) could be supported. Actually, each data provider should pre-process his data, so called, generalization. We briefly introduce generalization here.

Definition 5. *Generalization*
Generalization is defined by a function $\Phi = \{\Phi_1, \Phi_2, \cdots, \Phi_d\}$, *where* $\Phi_i : v \rightarrow p$ *maps each value* $v \in \Omega(A_i)$ *to a* $p \in P(A_i)$.

Let $D = \{r_1, \cdots, r_n\}$ be a multiset of records, where each record r_i represents the information of an individual with d attributes $A = \{A_1, \cdots, A_d\}$. We represent the data set D in a tabular form. We assume that each attribute A_i has a finite domain, denoted by $\Omega(A_i)$. The domain of D is defined as $\Omega(D) = \Omega(A_1)\times, \cdots, \times\Omega(A_d)$. To anonymize a dataset D, generalization replaces a value of an attribute with a more general value. The exact general value is determined according to the attribute partition.

Definition 6. *Attribute Partition*
The partitions $P(A_i)$ *of a numerical attribute are the intervals* $\langle I_1, I_2, \cdots, I_k\rangle$ *in* $\Omega(A_i)$ *such that* $\bigcup_{j=1}^{k} I_j = \Omega(A_i)$. *For categorical attribute, partitions are defined by a set of nodes from the taxonomy tree such that it covers the whole tree, and each leaf node belongs to exactly one partition.*

We give an example of data generalization in Fig. 3. Data provider will construct different layers of his data, which should match to different type of query. If the query function f is about the age, CSP could just add noise on layer y instead of layer z.

Alternatively, as the straightforward method will cost a lot of storage space and bandwidth, we are looking another way to solve such problems. Inspired by Mohammed's method [15], we propose our scheme (Fig. 2) that consists of our secure building blocks. The encryption $[\cdot]$ here is AHE scheme which is more efficient than FHE schemes.

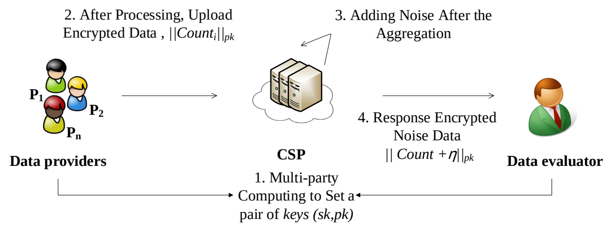

2. After Processing, Upload Encrypted Data , $||Count_i||_{pk}$
3. Adding Noise After the Aggregation
P1
P2
Pn
Data providers
CSP
4. Response Encrypted Noise Data $|| Count + \eta||_{pk}$
Data evaluator
1. Multi-party Computing to Set a pair of keys (sk,pk)

Figure 2: Using additive homomorphic encryption to achieve outsourced differential privacy

The building blocks of our scheme could be summarized as follows:

- **Setup**. Suppose that there are k data providers and one data evaluator in the system, denoted by $\mathrm{P}_1, \mathrm{P}_2, \cdots, \mathrm{P}_k$ and P_e, respectively. In this step, each data provider/evaluator together to generate a pair of keys (sk, pk) using secure multi-party computing.
- **Data uploading**. In this step, data providers will pre-process their data before uploading. First, data providers generalize their data according to the Attribute Partition. Then each data provider encrypts his group count $Count_i$ using the public key $Enc(pk, Count_i)$. After encryption, the ciphertext $[[Count_i]]$ is sent to the cloud server.
- **Noise addition**. Firstly, CSP aggregates all data providers' group count as $[[Count]]$. Then CSP gets the query function f from DE and choose the proper layer of group count to add noise on. Then the cloud server generates the noise η and uses the $Add([[Count]], [[\eta]])$ algorithm to add noise to the chosen layer of data providers' group counts. After the noise addition, the CSP sends $[[Count+\eta]]$ to DE.
- **Data analysis**. The data evaluator uses $Dec([[Count + \eta]], sk)$ to obtain the noisy count $Count' = Count + \eta$ and do some machine learning tasks.

The details of the procedures are shown in Algorithm 1.

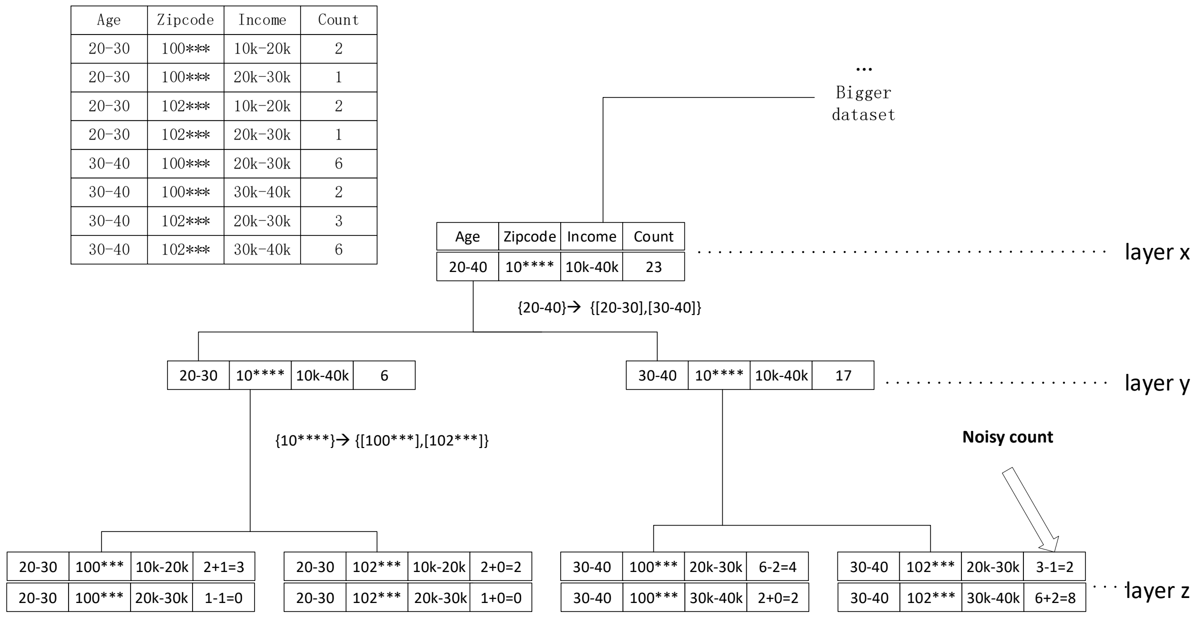
Algorithm 1 Outsourced differential privacy scheme based on secure building blocks

```
Input: Data providers (P): clean datasets $M_1, M_2, \cdots, M_k$;
       Data evaluator (DE): query function $f$;
Output: DE: group count with noise $Count'$;
 1: P$_i$ & DE: set a pair of keys $\{sk, pk\}$;
 2: for each $i \in [1, \cdots, k]$ do
 3:     P$_i$: generalize the dataset $Count_i \leftarrow G(M_i)$;
 4:     P$_i$: send the ciphertext $[[Count_i]]_{pk}$ to CSP;
 5: end for
 6: CSP: aggregate all data providers' data $[[Count]]_{pk} \leftarrow \Sigma_{i=1}^{k}[[Count_i]]_{pk}$;
 7: CSP: send $f$ to DE;
 8: CSP: choose the suitable layer of data $\in [[Count]]_{pk}$ depends on $f$;
 9: CSP: calculate the global sensitivity $\Delta f$;
10: CSP: generate the noise $\eta \sim \mathcal{X}(\Delta f, \epsilon)$, where $\mathcal{X}$ is a type of distribution;
11: CSP: calculate $[[Count + \eta]]_{pk}$;
12: CSP: send $||Count + \eta||_{pk}$ to DE;
13: DE: decrypt the ciphertext to get noised data $Count' \leftarrow Dec([[Count + \eta]]_{pk}, sk)$;
14: return $Count'$.
```

4.3 An Advanced Scheme Supporting Multiple Evaluators

Suppose there are more than one data evaluator asking for data, what should those schemes mentioned above to deal with. Well, in straight-forward scheme and our scheme using additive homomorphic, the more data evaluators there are, the more encrypted data with different key there should be. Obviously, CSP, who is running these schemes, would consume a lot of space for data storage; data providers, whose data are outsourced to CSP, must spend more computing power and time on his data uploading process (pre-processing for different data evaluator, communication with each data evaluator to settle different key for data encryption). To support more data evaluators, we designed our scheme for differentially private data publishing with SGX, a trustworthy component for secure computing, see Fig.4.

All operations about data providers' secure data was conducted in this component, no plain data should be obtained outside SGX. The CSP only handles the transmitting functions. In this way, even the CSP and data evaluators are colluded, the data security can be guaranteed, which is a very import property of our scheme compared to the others above. Before contributing data, each provider P_i encrypts



Age	Zipcode	Income	Count
20-30	100***	10k-20k	2
20-30	100***	20k-30k	1
20-30	102***	10k-20k	2
20-30	102***	20k-30k	1
30-40	100***	20k-30k	6
30-40	100***	30k-40k	2
30-40	102***	20k-30k	3
30-40	102***	30k-40k	6

...
Bigger dataset
Age | Zipcode | Income | Count
20-40 | 10**** | 10k-40k | 23
layer x
{20-40}→ {[20-30],[30-40]}
20-30 | 10**** | 10k-40k | 6
30-40 | 10**** | 10k-40k | 17
layer y
{10****}→ {[100***],[102***]}
Noisy count
20-30 | 100*** | 10k-20k | 2+1=3
20-30 | 100*** | 20k-30k | 1-1=0
20-30 | 102*** | 10k-20k | 2+0=2
20-30 | 102*** | 20k-30k | 1+0=0
30-40 | 100*** | 20k-30k | 6-2=4
30-40 | 100*** | 30k-40k | 2+0=2
30-40 | 102*** | 20k-30k | 3-1=2
30-40 | 102*** | 30k-40k | 6+2=8
layer z

Figure 3: An example for data processing (Generalization)

its data with its own key k_i by using a symmetric encryption scheme $[[[\cdot]]]$. The details of the procedure are shown in Algorithm 2.

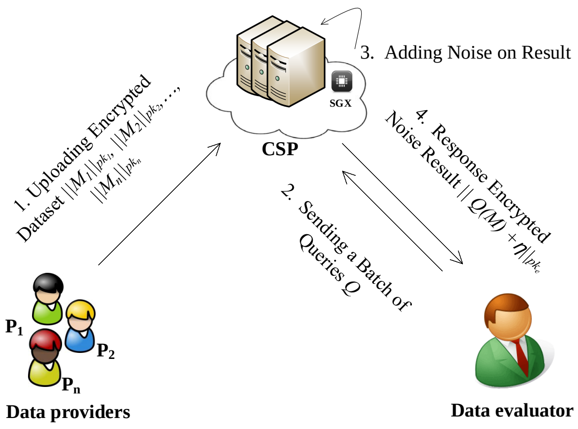

3. Adding Noise on Result
SGX
CSP
1. Uploading Encrypted Dataset $||M_1||_{pk_1}, ||M_2||_{pk_2}, \ldots, ||M_n||_{pk_n}$
2. Sending a Batch of Queries Q
4. Response Encrypted Noise Result $||Q(M)+\eta||_{pk_e}$
P1
P2
Pn
Data providers
Data evaluator

Figure 4: Using SGX in scheme to support multiple data evaluators

In this scheme, we use symmetric encryption instead of public key encryption due to the fully trusted property of SGX. With SGX's help, CSP could decrypt the ciphertext of data providers' data and perform calculations over this plaintext, which will save CSP a lot of time on operation. Also, because of the result is in plaintext form, CSP could use any data evaluator's symmetric key k_i to encrypt it. This make our scheme differs from previous schemes that the result is encrypted by specific data evaluator's public key and could only support one data evaluator. The evaluator will finally get the noised query result $r' = f(M) + \eta$. So, the scheme is not only efficient, but also suitable for multiple data evaluators. The storage security based on symmetric encryption. The security of data privacy based on the technique of SGX.

Algorithm 2 Outsourced differential privacy scheme supporting multiple data evaluators

```
Input: Data provider (P): clean datasets $M_1, M_2, \cdots, M_k$;
    Data evaluator (DE): query function $f$;
Output: DE: noisy result $r'$;
 1: for each $i \in [1, \cdots, k]$ do
 2:     P$_i$: communicate with SGX of P$_i$ to settle the symmetric key $k_i$;
 3:     P$_i$: encrypt $M_i$ to ciphertext $[[[M_i]]]_{k_i}$;
 4:     P$_i$: send $[[[M_i]]]_{k_i}$ to CSP;
 5: end for
 6: DE: communicate with SGX of CSP to settle the symmetric key $k_e$;
 7: CSP: communicate with DE and each P$_i$ to get privacy budget $\epsilon$;
 8: DE: send query $f$ to CSP;
 9: CSP: calculate the aggregation result $M \leftarrow \Sigma_{i=1}^{k} Dec([[[M_i]]]_{k_i}, k_i)$ in SGX of CSP;
10: CSP: calculate the query result $r \leftarrow Q(M)$ in SGX of CSP;
11: CSP: calculate $f$'s global sensitivity $\Delta f \leftarrow \max_{m,m'} \|f(m) - f(m')\|_1$ in SGX of CSP, where $m$ and $m'$ are any two records in the global set;
12: CSP: generate noisy result $r' \leftarrow r + \eta$ use parameters $(\Delta f, \epsilon)$ in SGX of CSP;
13: CSP: encrypt the noisy result to ciphertext $[[[r']]]_{k_e}$;
14: CSP: send $[[[r']]]_{k_e}$ to DE;
15: DE: decrypt the ciphertext to get noised data $r' \leftarrow Dec([[[Q(M) + \eta]]]_{k_e}, k_e)$;
16: return $r'$.
```

5 Security Analysis

Clearly, our goals are to protect the data providers' data from any adversary (ADV). In this section, we only present the security proof about data transmission process and noise addition process. Note that, for the data storage process and decryption process, since they depend on the encryption algorithm applied, we will not discuss the security for the two processes here.

The data flows are directed from the data provider to the CSP and from the CSP to the data evaluator. Generally speaking, an ADV may eavesdrop on the communication flow from the data provider to the CSP rather than the data from CSP to DE, which are often with noise and convinced to be privacy-leakage-proof. However, even if the ADV obtains a message in this way he cannot learn anything from this message without the data evaluator's secret key sk. The data's security depends on the encryption algorithm used (e.g., Paillier). Therefore, we can assert that the ADV cannot access private data even if the communication flow is intercepted. Notice that, in our multi-data-evaluator supporting scheme, data sending from data providers to CSP are encrypted by the key owned by themselves and SGX. CSP could even collude with data evaluators and any of them cannot reveal the private data of any data providers due to the lack of symmetric key.

Also, all the operations during noise addition procedure that have access to plaintext were conducted in SGX and SGX could be fully trusted to keep anyone from the data inside itself. So the data privacy during noise addition could be guaranteed.

6 Evaluation

In this section, we evaluate the performance of our schemes in terms of functionality, computational overhead and communication overhead. All experiments were conducted on a PC with a 3.40 GHz Intel I7-6700 CPU with Radeon(TM) R5 240 and 8 GB of RAM.

6.1 Scheme Supporting Multiple Data Evaluators with AHE

This scheme uses AHE to support non-interactive data publishing, which could help data providers go offline and save a lot of storage cost. So we will evaluate its functionality to prove it is effective; evaluate its computational overhead to prove it is efficient. It is obvious that data providers could safe a lot storage space for their data storage once they go offline, so we will not evaluate its communication overhead.

6.1.1 Functionality

We used datasets acquired from the UCI machine learning repository, which can be downloaded from UCI[1], to evaluate our scheme's functionality. We reserved $\frac{1}{10}$ of each dataset to serve as a test dataset, and we chose $\epsilon = 0.1$ as our privacy budget. For simplicity, we use Laplace mechanism to generate noises.

Definition 7. *(Laplace mechanism)*
Let m be a record in database M ($m \in M$), and let η be a random variable such that $\eta \sim Lap(\Delta f/\epsilon)$. The Laplace mechanism is defined as follows:

$$f'(m) = f(m) + \eta. \tag{5}$$

1. http://archive.ics.uci.edu/ml/

In the aspect of publishing data via non-interaction framework, we mainly care about the data utility. Therefore, we compare the accuracy of the two classification results. One is the data published by scheme using additive homomorphic encryption. The other one is the original data. Fig. 5(a) shows the accuracies of training KNN and Naive Bayes classifiers for Adult, Letter Recognition, CPU, Glass and EEG Eye State. From the figure, we can see that training a classifier on the noisy data instead of the original data exerts little influence on the classifier performance, which means our scheme is feasible.

6.1.2 Computational Overhead

We use Paillier as our additive homomorphic encryption algorithm in non-interactive publishing scheme.

As seen in Fig. 5(b), each enc/dec operation takes 30ms in Paillier, far less than 500ms in FHE. Moreover, it takes only 48ms to perform 1000 addition operations on ciphertext using Paillier. Most of the operations in our scheme is noise addition, so our scheme is efficient enough to be applied.

6.2 Scheme Supporting Multiple Data Evaluators with SGX

6.2.1 Functionality

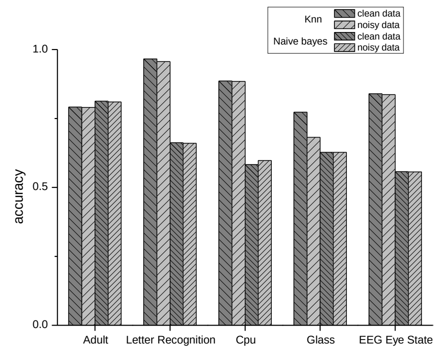
The scheme's functionality can be seen in Fig. 6(a), its accuracy is similar to the scheme using additive homomorphic encryption.

6.2.2 Computational Overhead

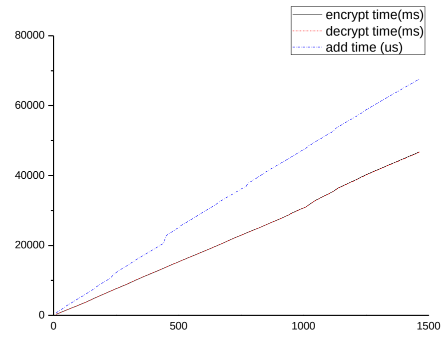
We choose AES-256 as our symmetric encryption algorithm, running in SGX's enclave. All the data are decrypted in SGX's enclave to build a larger dataset (Adult, 45,222 census records) for calculating the result to all the queries from data evaluators. It only takes milliseconds to have our noise generated and added. So we care more about the I/O and enc/dec speed in SGX. As seen in Fig. 6(b), AES-256 only costs 80ms to perform 1000 enc/dec operations (faster than Paillier). And it takes 4s to process the whole Adult dataset, see Fig. 6(c), which is more quicker than straight-forward scheme using FHE. Because the computational overhead is low enough, our scheme is acceptable and feasible.

6.2.3 Communication Overhead

Our scheme include two phases that incur communication costs, including data uploading and data analysis. It cost only Bytes to send a result from CSP to DE, so we only check the communication overhead during data uploading. In the data uploading phase (see Fig 6(d)), the size of the message sent from data providers to CSP is growing linear with the increase of dataset itself.

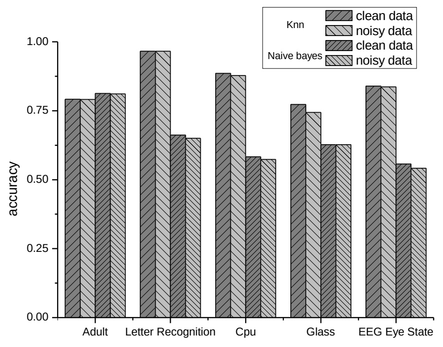


(a) Functionality Using Paillier

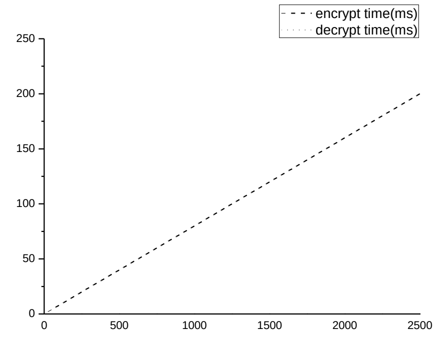


(b) Computational Overhead Using Paillier

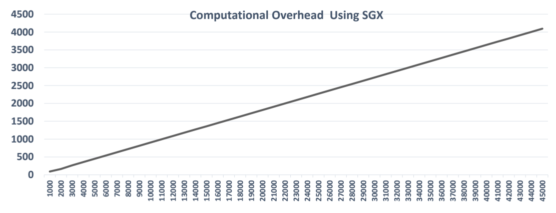
Figure 5: Performance of Basic Scheme



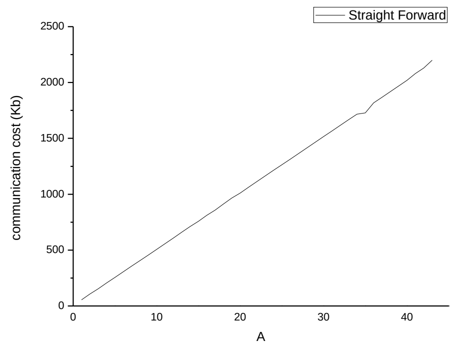
(a) Functionality Using AES-256 in SGX



(b) Computational Overhead Using AES-256



(c) Computational Overhead Using SGX



(d) Communication Overhead

Figure 6: Performance of Advance Scheme

7 Conclusions

In this paper, we addressed the issues of inefficiency due to implementation of different privacy mechanisms for differentially private publishing. Specifically, we proposed our efficient and secure outsourced differential privacy schemes. The one scheme makes secure outsourcing publishing more efficient by using our preprocessing method and secure building blocks. The other scheme expands the secure data publishing to a multi-evaluator scenario. In both schemes, the data providers are not required to be on-line when the query on their data is requested. The experiment showed that the efficiency of our new scheme compared with the basic solutions. In future work, we will consider optimize the algorithm used in SGX to boost efficiency. In addition, we are also interested in researching special encryption algorithms that permit smart and efficient implementation of privacy mechanisms.

Acknowledgement

Jin Li is supported by National Natural Science Foundation of China for Outstanding Youth Foundation (No. 61722203) and National Natural Science Foundation of China for

Joint Fund Project (No. U1936218). Tong Li is supported by National Natural Science Foundation of China (No. 61802078 and No. 61702125). The work of Lou and Hou was supported in part by Virginia Commonwealth Cyber Initiative.

References

[1] Jin Li, Heng Ye, Wei Wang, Wenjing Lou, Y Thomas Hou, Jiqiang Liu, and Rongxing Lu. Efficient and secure outsourcing of differentially private data publication. In *European Symposium on Research in Computer Security*, pages 187–206. Springer, 2018.
[2] Jaideep Vaidya and Chris Clifton. Privacy-preserving k-means clustering over vertically partitioned data. pages 206–215, 2003.
[3] Jaideep Vaidya and Chris Clifton. Privacy preserving association rule mining in vertically partitioned data. pages 639–644, 2002.
[4] Cynthia Dwork, Frank Mcsherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. 3876:265–284, 2006.
[5] Frank Mcsherry and Kunal Talwar. Mechanism design via differential privacy. pages 94–103, 2007.
[6] Frank Mcsherry. Privacy integrated queries: an extensible platform for privacy-preserving data analysis. *Communications of The ACM*, 53(9):89–97, 2010.
[7] Cynthia Dwork, Moni Naor, and Salil Vadhan. The privacy of the analyst and the power of the state. pages 400–409, 2012.
[8] Xiaokui Xiao, Guozhang Wang, and Johannes Gehrke. Differential privacy via wavelet transforms. *IEEE Transactions on Knowledge and Data Engineering*, 23(8):1200–1214, 2011.
[9] Chao Li, Michael Hay, Vibhor Rastogi, Gerome Miklau, and Andrew Mcgregor. Optimizing linear counting queries under differential privacy. pages 123–134, 2010.
[10] Ganzhao Yuan, Zhenjie Zhang, Marianne Winslett, Xiaokui Xiao, Yin Yang, and Zhifeng Hao. Low-rank mechanism: optimizing batch queries under differential privacy. *Proceedings of The Vldb Endowment*, 5(11):1352–1363, 2012.
[11] H V Jagadish, Nick Koudas, S Muthukrishnan, Viswanath Poosala, Kenneth C Sevcik, and Torsten Suel. Optimal histograms with quality guarantees. *very large data bases*, pages 275–286, 2010.
[12] Michael Hay, Vibhor Rastogi, Gerome Miklau, and Dan Suciu. Boosting the accuracy of differentially private histograms through consistency. *Proceedings of The Vldb Endowment*, 3:1021–1032, 2010.
[13] Yonghui Xiao, Li Xiong, and Chun Yuan. Differentially private data release through multidimensional partitioning. pages 150–168, 2010.
[14] Jia Xu, Zhenjie Zhang, Xiaokui Xiao, Yin Yang, Ge Yu, and Marianne Winslett. Differentially private histogram publication. *very large data bases*, 22(6):797–822, 2013.
[15] Noman Mohammed, Rui Chen, Benjamin C M Fung, and Philip S Yu. Differentially private data release for data mining. pages 493–501, 2011.
[16] Rui Chen, Noman Mohammed, Benjamin C M Fung, Bipin C Desai, and Li Xiong. Publishing set-valued data via differential privacy. *Proceedings of The Vldb Endowment*, 2011.
[17] Graham Cormode, Cecilia M Procopiuc, Divesh Srivastava, Entong Shen, and Ting Yu. Differentially private spatial decompositions. pages 20–31, 2012.
[18] Wahbeh Qardaji, Weining Yang, and Ninghui Li. Differentially private grids for geospatial data. pages 757–768, 2013.
[19] Boaz Barak, Kamalika Chaudhuri, Cynthia Dwork, Satyen Kale, Frank Mcsherry, and Kunal Talwar. Privacy, accuracy, and consistency too: a holistic solution to contingency table release. pages 273–282, 2007.
[20] Shiva Prasad Kasiviswanathan, Mark Rudelson, Adam Smith, and Jonathan Ullman. The price of privately releasing contingency tables and the spectra of random matrices with correlated rows. pages 775–784, 2010.
[21] Cynthia Dwork, Moni Naor, Omer Reingold, Guy N Rothblum, and Salil Vadhan. On the complexity of differentially private data release: efficient algorithms and hardness results. pages 381–390, 2009.
[22] Cynthia Dwork and Jing Lei. Differential privacy and robust statistics. pages 371–380, 2009.
[23] Moritz Hardt, Guy N Rothblum, and Rocco A Servedio. Private data release via learning thresholds. pages 168–187, 2012.
[24] Cynthia Dwork, Guy N Rothblum, and Salil Vadhan. Boosting and differential privacy. pages 51–60, 2010.
[25] Wahbeh Qardaji, Weining Yang, and Ninghui Li. Priview: practical differentially private release of marginal contingency tables. pages 1435–1446, 2014.
[26] Shiva Prasad Kasiviswanathan, Homin K Lee, Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. What can we learn privately. pages 531–540, 2008.
[27] Avrim Blum, Katrina Ligett, and Aaron Roth. A learning theory approach to non-interactive database privacy. pages 609–618, 2008.
[28] Manas A Pathak, Shantanu Rane, and Bhiksha Raj. Multiparty differential privacy via aggregation of locally trained classifiers. pages 1876–1884, 2010.
[29] Sunil Gupta, Santu Rana, and Svetha Venkatesh. Differentially private multi-task learning. pages 101–113, 2016.
[30] Reza Shokri and Vitaly Shmatikov. Privacy-preserving deep learning. pages 1310–1321, 2015.
[31] Zhanglong Ji, Xiaoqian Jiang, Shuang Wang, Li Xiong, and Lucila Ohnomachado. Differentially private distributed logistic regression using private and public data. *BMC Medical Genomics*, 7(1):1–10, 2014.
[32] Sen Su, Peng Tang, Xiang Cheng, Rui Chen, and Zequn Wu. Differentially private multi-party high-dimensional data publishing. pages 205–216, 2016.
[33] David Chaum and Torben Pedersen. Wallet databases with observers. pages 89–105, 1992.
[34] Benoit Chevalliermames, Jeansebastien Coron, Noel Mccullagh, David Naccache, and Michael L Scott. Secure delegation of elliptic-curve pairing. 6035:24–35, 2010.
[35] David Benjamin and Mikhail J Atallah. Private and cheating-free outsourcing of algebraic computations. pages 240–245, 2008.
[36] Ping Li, Jin Li, Zhengan Huang, Tong Li, Chongzhi Gao, Xiuming Liu, and Kai Chen. Multi-key privacy-preserving deep learning in cloud computing. *Future Generation Computer Systems*, 74:76–85, 2017.
[37] Ping Li, Jin Li, Zhengan Huang, Chongzhi Gao, Wenbin Chen, and Kai Chen. Privacy-preserving outsourced classification in cloud computing. cluster computing. *Cluster Computing*, pages 1–10, 2017.
[38] Florian Tramer and Dan Boneh. Slalom: Fast, verifiable and private execution of neural networks in trusted hardware. *arXiv preprint arXiv:1806.03287*, 2018.
[39] Andrew Baumann, Marcus Peinado, and Galen C Hunt. Shielding applications from an untrusted cloud with haven. *ACM Transactions on Computer Systems*, 33(3):8, 2015.
[40] Shweta Shinde, DL Tien, Shruti Tople, and Prateek Saxena. Panoply: Low-tcb linux applications with sgx enclaves. In *Proceedings of the Annual Network and Distributed System Security Symposium (NDSS)*, page 12, 2017.
[41] Felix Schuster, Manuel Costa, Cédric Fournet, Christos Gkantsidis, Marcus Peinado, Gloria Mainar-Ruiz, and Mark Russinovich. Vc3: Trustworthy data analytics in the cloud using sgx. In *Security and Privacy (SP), 2015 IEEE Symposium on*, pages 38–54. IEEE, 2015.
[42] Stefan Brenner, Colin Wulf, David Goltzsche, Nico Weichbrodt, Matthias Lorenz, Christof Fetzer, Peter R Pietzuch, and Rudiger Kapitza. Securekeeper: Confidential zookeeper using intel sgx. page 14, 2016.
[43] Tyler Hunt, Zhiting Zhu, Yuanzhong Xu, Simon Peter, and Emmett Witchel. Ryoan: A distributed sandbox for untrusted computation on secret data. In *OSDI*, pages 533–549, 2016.
[44] Olga Ohrimenko, Felix Schuster, Cédric Fournet, Aastha Mehta, Sebastian Nowozin, Kapil Vaswani, and Manuel Costa. Oblivious multi-party machine learning on trusted processors. In *USENIX Security Symposium*, pages 619–636, 2016.

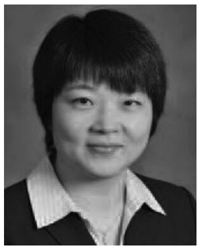
This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2020.3015886, IEEE Transactions on Dependable and Secure Computing

Jin Li is currently a professor and vice dean of School of Computer Science, Guangzhou University. He received his B.S. (2002) and M.S. (2004) from Southwest University and Sun Yat-sen University, both in Mathematics. He got his Ph.D degree in information security from Sun Yat-sen University at 2007. His research interests include design of secure protocols in Cloud Computing and cryptographic protocols. He has published more than 100 papers in international conferences and journals, including IEEE INFOCOM, IEEE TIFS, IEEE TPDS, IEEE TOC and ESORICS etc. His work has been cited more than 10000 times at Google Scholar and the H-Index is 34. He also served as program chairs and committee for many international conferences. He received NSFC Outstanding Youth Foundation in 2017.

Heng Ye is a Ph.D. candidate at Beijing Jiaotong University since 2016. His research interests include differential privacy , attribute based encryption and IOT.

Tong Li received his B.S. and M.S. from Taiyuan University of Technology and Beijing University of Technology, in 2011 and 2014, respectively, both in Computer Science & Technology. He got his Ph.D degree in information security from Nankai University at 2017. Currently, he is a post-doctoral research at Guangzhou University. His research interests include applied cryptography and data privacy protection in cloud computing.

Wei Wang is currently a full professor and chairs the Department of Information Security, Beijing Jiaotong University, China. He earned his Ph.D. degree in control science and engineering under the supervision of Prof. Xiaohong Guan from Xian Jiaotong University, in 2006. He was a postdoctoral researcher in University of Trento, Italy, during 2005-2006. He was a postdoctoral researcher in TELECOM Bretagne and in INRIA, France, during 2007-2008. He was a European ERCIM Fellow in Norwegian University of Science and Technology (NTNU), Norway, and in Interdisciplinary Centre for Security, Reliability and Trust (SnT), University of Luxembourg, during 2009-2011. He visited INRIA, ETH, NTNU, CNR, and New York University Polytechnic. He has authored or co-authored over 80 peer-reviewed papers in various journals and international conferences. His main research interests include mobile, computer and network security.



Wenjing Lou received her Ph.D. in Electrical and Computer Engineering from the University of Florida. She joined the Electrical and Computer Engineering department at Worcester Polytechnic Institute as an assistant professor in 2003, where she was promoted to associate professor with tenure in 2009. In 2011, she joined the Computer Science department at Virginia Tech as an associate professor with tenure. Her current research interests are in the area of cyber security, with emphases on wireless network security, security and privacy in cloud computing and cyber physical systems. She is also interested in network protocols. She is currently serving on the editorial board of five journals: IEEE Transactions on Wireless Communications, IEEE Transactions on Smart Grid, IEEE Wireless Communications Letter, Elsevier Computer Networks, and Springer Wireless Networks. She has served as TPC co-chair for the security symposium of several leading IEEE conferences, including General Symposium at IEEE Globecom 2007, Network Security and Privacy Track at IEEE ICCCN 2009, Security Symposium at IEEE ICC 2010, Security and Localization Track at IEEE PIMRC 2011, and Security Symposium at IEEE Globecom 2012. She serves as TPC member regularly for many premier IEEE and ACM conferences.

Y. Thomas Hou received the Ph.D. degree from the NYU Tandon School of Engineering in 1998. During 1997 to 2002, he was a member of Research Staff with Fujitsu Laboratories of America, Sunnyvale, CA, USA. He is Bradley Distinguished Professor of electrical and computer engineering with Virginia Tech, Blacksburg, VA, USA, which he joined in 2002. His current research focuses on developing innovative solutions to complex science and engineering problems arising from wireless and mobile networks. He has authored or co-authored over 100 journal papers and 130 conference papers in networking related areas. His papers were recognized by five best paper awards from the IEEE and two paper awards from the ACM. He holds five U.S. patents. He authored/co-authored two graduate textbooks: Applied Optimization Methods for Wireless Networks (Cambridge University Press, 2014) and Cognitive Radio Communications and Networks: Principles and Practices (Academic Press/Elsevier, 2009). He is the Steering Committee Chair of the IEEE INFOCOM conference and a member of the IEEE Communications Society Board of Governors.

Jiqiang Liu received the Ph.D. degree in Beijing Normal University in 1999. He now works at Beijing Jiaotong University as a professor as well as Associate Dean of Graduate School. He is a IEEE fellow and has published more than 100 research papers. His research interests includes Trusted Computing, Privacy Preserving and Security Protocol.

Rongxing Lu has been an assistant professor at the Faculty of Computer Science (FCS), University of New Brunswick (UNB), Canada, since August 2016. Before that, he worked as an assistant professor at the School of Electrical and Electronic Engineering, Nanyang Technological University (NTU), Singapore from April 2013 to August 2016. Rongxing Lu worked as a Postdoctoral Fellow at the University of Waterloo from May 2012 to April 2013. He was awarded the most prestigious “Governor Generals Gold Medal”, when he received his PhD degree from the Department of Electrical and Computer Engineering, University of Waterloo, Canada, in 2012; and won the 8th IEEE Communications Society (ComSoc) Asia Pacific (AP) Outstanding Young Researcher Award, in 2013. He is presently a senior member of IEEE Communications Society. His research interests include applied cryptography, privacy enhancing technologies, and IoT-Big Data security and privacy. He has published extensively in his areas of expertise (with citation 14,900+ and H-index 60 from Google Scholar as of April 2019), and was the recipient of 8 best (student) paper awards from some reputable journals and conferences. Currently, Dr. Lu serves as the Vice-Chair (Publication) of IEEE ComSoc CIS-TC (Communications and Information Security Technical Committee). Dr. Lu is the Winner of 2016-17 Excellence in Teaching Award, FCS, UNB.