PRIVACY-PRESERVING DATA MINING

A Dissertation

by

NAN ZHANG

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

December 2006

Major Subject: Computer Science

PRIVACY-PRESERVING DATA MINING

A Dissertation

by

NAN ZHANG

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Approved by:

Co-Chairs of Committee,	Wei Zhao
	Jianer Chen
Committee Members,	Jennifer L. Welch
	Costas N. Georghiades
Head of Department,	Valerie E. Taylor

December 2006

Major Subject: Computer Science

ABSTRACT

Privacy-Preserving Data Mining. (December 2006)

Nan Zhang, B.S., Beijing University

Co-Chairs of Advisory Committee: Dr. Wei Zhao
Dr. Jianer Chen

In the research of privacy-preserving data mining, we address issues related to extracting knowledge from large amounts of data without violating the privacy of the data owners. In this study, we first introduce an integrated baseline architecture, design principles, and implementation techniques for privacy-preserving data mining systems. We then discuss the key components of privacy-preserving data mining systems which include three protocols: data collection, inference control, and information sharing. We present and compare strategies for realizing these protocols. Theoretical analysis and experimental evaluation show that our protocols can generate accurate data mining models while protecting the privacy of the data being mined.

ACKNOWLEDGEMENTS

I would like to thank my committee co-chairs, Dr. Zhao and Dr. Chen, and my committee members, Dr. Welch and Dr. Georghiades, for their guidance and support throughout the course of my Ph.D. studies. Thanks also to my colleagues and the department faculty and staff for making my time at Texas A&M University so enjoyable.

The work reported in this dissertation was supported in part by the National Science Foundation under Contracts 0081761, 0324988, 0329181, by the Defense Advanced Research Projects Agency under Contract F30602-99-1-0531, and by Texas A&M University under its Telecommunication and Information Task Force Program. Any opinions, findings, conclusions, and/or recommendations expressed in this dissertation, either expressed or implied, are those of the author and do not necessarily reflect the views of the sponsors listed above.

NOMENCLATURE

B2B	Business-to-Business
B2C	Business-to-Customer
HIPAA	Health Insurance Portability and Accountability Act
OLAP	Online Analytical Processing
SMC	Secure Multiparty Computation

TABLE OF CONTENTS

Page

CHAPTER Page

LIST OF FIGURES

LIST OF TABLES

CHAPTER I

INTRODUCTION

Data mining is the process of extracting knowledge from large amounts of data [29]. It has been widely and successfully used for more than ten years in various domains, such as marketing, weather forecasting, medical diagnostics, anti-terror measures, etc. Nonetheless, the challenge remains to conduct data mining over private data (e.g., health information) without violating the privacy of data owners (e.g., patients).

Privacy protection has become a necessary requirement in many data mining applications due to emerging privacy legislation and regulations, such as the U.S. Health Insurance Portability and Accountability Act (HIPAA) [30] and the European Union's Privacy Directive [21]. This dissertation seeks to design and compare strategies for protecting privacy in data mining.

I.1 Baseline Architecture

Data mining is usually carried out in multiple steps. First, the data being mined are collected from their sources, which we refer to as data providers. In many systems, data providers are physically distributed, forming the bottom tier of the baseline architecture of data mining systems, as shown in Figure 1. Data providers are the data owners, and are expected to submit their (private) data to the data warehouse server, which forms the middle tier of the architecture. For example, in an online survey system, the survey respondents are the data providers who submit their data to the survey analyzer, which holds the data warehouse server.

This dissertation follows the style of *IEEE Transactions on Knowledge and Data Engineering*.

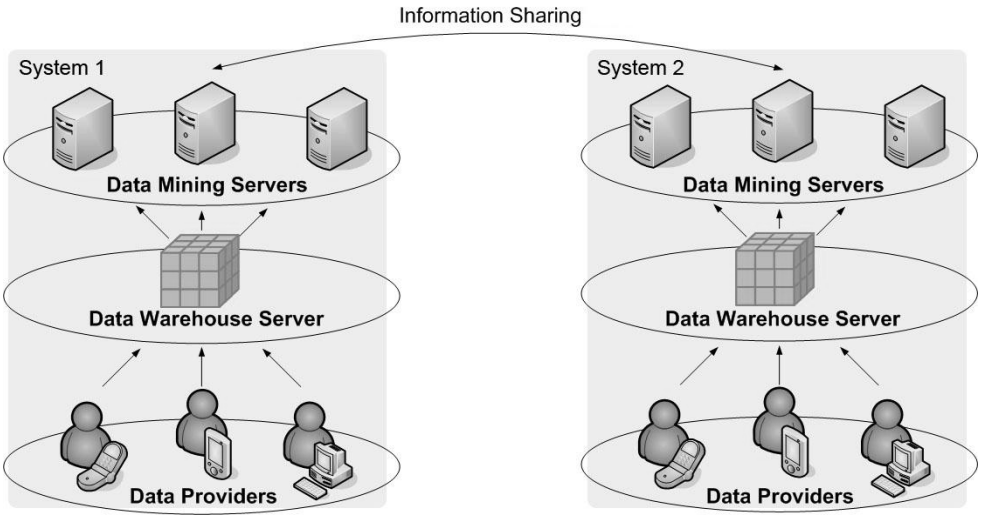

Fig. 1. Baseline Architecture.

In the data warehouse server, data collected from the data providers are stored in well-disciplined physical structures (e.g., multi-dimensional data cube), and are aggregated and pre-computed in various forms (e.g., sum, max, min). For example, in an online survey system, an aggregated data point may be the mean age of all survey respondents. The objective of data warehouse server is to support online analytical processing (OLAP) on the data, and to facilitate data mining [29].

The actual data mining tasks are performed by the data mining servers, which form the top tier of the baseline architecture. When performing data mining tasks, the data mining servers are likely to use the aggregated data, which are pre-computed by the data warehouse server, rather than the rough data, which are directly collected from the data providers, in order to hasten the data mining process.

Note that the data mining servers may not have the right to access all data stored in the data warehouse. For example, in a hospital where all patients' information is stored in

the data warehouse, the accounting department of the hospital (as a data mining server) is allowed to access patients' financial data, but is prohibited from accessing patients' medical records per HIPAA requirements [30].

Besides constructing data mining models on its local data warehouse server, a data mining server may also share information with data mining servers from other systems (i.e., with other data warehouses), in order to construct data mining models spanning multiple data warehouses. Since each data mining server holds the local data mining model of its own system, in the information sharing process, each data mining server is likely to share its local data mining model, rather than the raw data stored in the data warehouse, to build globally valid data mining models across multiple systems. For example, several retail companies may share their local data mining models on customer records in order to build a global data mining model on consumer behavior. Note that the local data mining models can be private and need to be protected, especially when these models are not valid globally.

I.2 Design Principle

In order to introduce the design principle of privacy-preserving data mining systems, we need to define the term "privacy". Privacy has been a central issue from a sociological standpoint [60]. In the context of information privacy, information is considered to be private if its owner has the *right to choose* whether or not, to what extent, and for what purpose, to disclose the information to others [48]. In the literature on privacy-preserving data mining, it is commonly (explicitly or tacitly) assumed that a data owner generally chooses not to disclose its private data unless the disclosure is necessary for

the purpose of data mining. Based on this assumption, we can state the design principle of privacy-preserving data mining systems as follows.

> *Minimum Necessary Rule*: Disclosed private information (from one entity to others) in a data mining system should be limited to the minimum necessary for data mining.

Note that the "*minimum*" here is a qualitative measure rather than a quantitative one. Since the quantitative measure of privacy disclosure varies between different systems and/or different data owners, we use the term "minimum" in the design principle to state that all private information unnecessary (or less necessary, as determined by the sensitivity of data and the accuracy requirements of data mining results) for data mining should *not* be disclosed in a privacy-preserving data mining system.

Due to the minimum necessary rule, the privacy disclosure in data mining systems should be allowed on a "need-to-know" (i.e., necessary-for-data mining) basis. The minimum necessary rule has been defined and mandated by privacy legislation and regulations. In particular, it is considered to be the key regulation of HIPAA privacy rules [30].

I.3 Basic Strategy

Based on the system architecture and design principle, we now introduce the basic design strategies for privacy-preserving data mining systems. Apparently, in a data mining system, privacy disclosure can occur when private data are transmitted from one entity to another. Thus, a commonly used privacy protection measure is to enforce privacy-preserving communication protocols between different entities, such that each

entity may follow the protocol and thereby prevent private information disclosure during data communication.

Specifically, three kinds of protocols are needed:

- *Data Collection Protocol*, which protects privacy during data transmission from the data providers to the data warehouse server;
- *Inference Control Protocol*, which manages the privacy protection between the data warehouse server and data mining servers;
- *Information Sharing Protocol*, which controls the information shared between different data mining servers (of different systems).

Based on the minimum necessary rule, a common goal of these protocols is to transmit from one entity to another the minimum amount of private information necessary to construct accurate data mining models. In reality, one may have to make a tradeoff between privacy protection and accuracy of constructed data mining models.

Note that although these three protocols have unique characteristics and can be used in different scenarios, not all systems require all three protocols to protect private information. We will analyze which systems require different combination of the three protocols while studying the integration of these protocols.

I.4 Dissertation Organization

The rest of this dissertation is organized as follows. In the second chapter, we briefly review the related work in privacy-preserving data mining. Then, we address the design of the three protocols. We propose a new scheme on data collection protocol in Chapter III. We introduce a cardinality-based inference control protocol in Chapter IV. In

Chapter V, we present the adversary models and design strategies of information sharing protocols. We address the effective integration of these three protocols in Chapter VI, and conclude with final remarks in Chapter VII.

CHAPTER II

RELATED WORK

There has been a growing amount of research in the area of privacy-preserving data mining. In this chapter, we briefly review related work on data collection protocol, inference control protocol, and information sharing protocol, respectively. Before that, we remark that the readers should not mistake our review in this chapter as an indicator that practical privacy-preserving data mining systems have been well developed and widely used. In fact, although there is ongoing work on the development of real privacy-preserving data mining systems [34], most work reviewed in this chapter presents proposals for privacy-preserving algorithms, rather than solutions to real system-building problems.

II.1 Data Collection Protocol

There are three kinds of approaches that have been proposed for data collection protocol: data exchange approach [12], noise insertion approach [1], [4], [6], [17], [19], [24], [47], and cryptographic approach [62].

When the data exchange approach is used, each data provider exchanges its data with another data provider before transmitting the data to the data warehouse server. As such, the data warehouse server does not know the real owners of the collected data. Nonetheless, the data collected by the data warehouse server are still able to support construction of data mining models.

The data exchange approach divulges the private information of one data provider to (at least) another data provider. Thus, this approach can only be used in systems where

every data provider is trustworthy. That is, no data provider has the intent to compromise the private information of another data provider. Note that in many practical systems (e.g., online survey), the data providers are untrustworthy (i.e., one data provider may intend to compromise the private data of another data provider). Apparently, the data exchange approach cannot protect the private information of data providers from being compromised (by other data providers) in these systems.

When the noise insertion approach is used, the data providers (independently) insert noise into their data, and transmit the perturbed data to the data warehouse server [1], [4], [6], [17], [19], [24], [47]. Noise can be inserted in two ways:

- by directly adding random noise to the original data values (e.g., change age from 23 to 30, change location from Texas to California), and
- by generalizing data values based on the corresponding domain hierarchy (e.g., generalize age from 23 to range 21-25, generalize location from Texas to USA).

The first method is also referred to as *perturbation-based method* in [56]. It can be applied to arbitrary data. The second method is referred to as *aggregation-based method* in [56]. This method can only be applied to data with a domain hierarchy known by the data warehouse server. Nonetheless, it can be used to guarantee *k*-anonymity [9] (i.e., the perturbed value of a data record is indistinguishable from the perturbed values of at least $k - 1$ other data records).

With the noise insertion approach, both the data providers and the data warehouse server know how the noise is inserted into the data (e.g., the distribution of the random noise, the generalization of domain hierarchy). The effectiveness of noise insertion

approach is based on two assumptions: 1) due to the inserted noise, it will be difficult, if not impossible, for the data warehouse server to recover the original private data from the perturbed data, and 2) the data warehouse server can still reconstruct the original data distribution based on the perturbed data and the knowledge of noise insertion approach, thereby supporting the construction of accurate data mining models.

Generally speaking, from a data provider's point of view, the noise insertion approach is independent of data mining tasks. Thus, the noise insertion approach can be used in scenarios in which there are multiple data mining tasks, or for which the data mining task is unknown at the time of data collection. A problem of the noise insertion approach, however, is that the perturbed data may still contain a substantial amount of private information. For example, researchers have found that the spectral structure of perturbed data can help the data warehouse server to compromise private data by separating noise from the original data [31], [38]. As such, the randomization approach cannot be used in systems where strict private protection needs to be enforced (e.g., the private data are either considerably sensitive or protected by privacy laws).

The cryptographic approach assumes that each data provider has a pair of public and private keys, and the data warehouse server knows the sum of the public keys of all data providers as pre-knowledge [62]. When the cryptographic approach is used, the data warehouse server first sends the sum of the public keys to all data providers as a reference. The data providers encrypt their data based on the reference and transmit the encrypted data to the data warehouse server. Based on mathematical manipulations, the

TABLE 1

Comparison of Related Work on Data Collection Protocol

Support	*Malicious Data Providers*	*Strict Privacy Protection*	*Open System*	*Undetermined Tasks*
Data Exchange	No	Yes	Yes	Yes
Noise Insertion	Yes	No	Yes	Yes
Cryptographic	No	Yes	No	No

data warehouse server can construct accurate data mining models based on the encrypted data, without knowing the values of the original data.

Since the cryptographic approach assumes that the data warehouse server has pre-knowledge about the sum of the public keys of all data providers, the cryptographic approach can only be used in *closed systems* where the identities of data providers are known by the data warehouse server before the data collection begins. Note that many practical systems (e.g., online survey) are *open* in that the data warehouse server (e.g., the survey collector) cannot know the identities of, or even the number of, data providers (e.g., which people will respond to the survey) until the completion of data collection. The cryptographic approach cannot be used in such open systems. Besides, the cryptographic approach requires the data providers to be trustworthy, requiring all data providers to properly follow the protocol and send correct inputs to the data warehouse server. If one data provider deviates from the designated protocol and submits incorrect input, the data mining results can be entirely destroyed.

Table 1 shows the comparison between the data exchange approach, noise insertion approach, and cryptographic approach. In particular, we show which kinds of systems can, or cannot, be supported by each approach.

II.2 Inference Control Protocol

There are two kinds of approaches that have been proposed for inference control protocol, namely the query-oriented approach [57] and the data-oriented approach [5].

To describe the query-oriented approach, we first need to introduce a concept called "*safe*" query set. A set of queries $\{ Q_1, Q_2, \ldots, Q_n \}$ is safe if a data mining server cannot infer private information from the answers to $Q_1, Q_2, \ldots, Q_n$. With this concept, the basic idea of query-oriented inference control approach can be easily described as follows. Upon receiving a query from a data mining server, the data warehouse server will answer the query if and only if the union set of query history (i.e., the set of all the queries already answered) and the recently received query is safe. Otherwise, the data warehouse server will reject the query.

Query-oriented inference control has also been extensively used in statistical databases [15]. The major difference between these systems and privacy-preserving data mining systems is that privacy-preserving data mining systems usually deal with larger amounts of data in a timely manner (recall that OLAP means *online* analytical processing). Therefore, inference control protocols in privacy-preserving data mining systems require more efficiency to conduct query processing.

Dynamically determining the safety of a query set can be time-consuming. Therefore, previous work focuses on a cardinality-based method, which is a static version of the query-oriented approach. With this method, a safe set of queries is determined *a prior* based on the cardinality of the data (e.g., how many data points are private). Apparently, any subset of this safe query set is also safe. At run-time, upon receiving a query, the

data warehouse server will answer the query if and only if the query is in the pre-determined safe set. Otherwise, the data warehouse server will reject the query.

Apparently, the cardinality-based query-oriented approach is conservative and may unnecessarily reject a large number of queries. Thus, it cannot be used in systems in which data mining servers require a high level of information availability (i.e., percentage of queries answered by the data warehouse server) to generate accurate data mining results.

When the data-oriented approach is used, the data stored in the data warehouse server are *perturbed*, and the query answers are estimated (as accurately as possible) based on the perturbed data. In certain cases (where the data collection protocol adopts the noise-insertion approach), the data perturbation is done by the data collection protocol. In other cases (where the original data need to be stored in the data warehouse server), the perturbation must be done at run-time when the query is processed. In either case, the specific perturbation techniques are similar to those used in the noise-insertion approach for data collection protocol. The effectiveness of data-oriented approach is based on two assumptions: 1) the perturbation is enough to prevent private information from being disclosed, and 2) the query answers estimated from the perturbed data can still support construction of accurate data mining models.

Similar to the noise-insertion approach for data collection protocol, a problem of the data-oriented approach is that the perturbed data may still contain a substantial amount of private information. As such, the data-oriented approach cannot be used in systems where private-protection guarantee is required by privacy laws.

TABLE 2

Comparison of Related Work on Inference Control Protocol

Support	*High Efficiency*	*High Data Availability*	*Sensitive Data*	*Perturbed Data*
Cardinality-based	Yes	No	Yes	No
Dynamic Query-Oriented	No	Yes	Yes	No
Data-Oriented	Yes	Yes	No	Yes

Table 2 shows the comparison between cardinality-based (i.e., static) query-oriented approach, dynamic query-oriented approach, and data-oriented approach. In particular, we show which kinds of systems can, or cannot, be supported by each approach.

II.3 Information Sharing Protocol

Most existing work on information sharing protocol [2], [3], [16], [18], [25], [32], [36], [37], [39], [41], [53], [54], [55] considers the privacy-preserving information sharing problem as a variation of the secure multiparty computation (SMC) problem [27], and use cryptographic approaches to solve the problem.

Since it is difficult to achieve security against adversaries with unrestricted behavior in SMC [27], most existing information sharing protocols make restrictions on the behavior of adversaries. There are two kinds of restrictions that are commonly employed: 1) *semi-honest* (i.e., honest-but-curious) restriction, which assumes that all adversaries properly follow the protocol, with the only exception being that the adversaries may record all intermediate computation and communication [2], [3], [16], [18], [32], [36], [37], [41], [53], [54], [55], [61], and 2) *malicious* restriction, which assumes that an adversary may deviate from the protocol, but cannot change its input [25], [39]. With either restriction, the basic idea of cryptographic approach is for each participating data

TABLE 3

Comparison of Related Work on Information Sharing Protocol

Support	*High Efficiency*	*Adversaries that Deviate from the protocol*	*Adversaries that Change Input*
Semi-honest Restriction	Yes	No	No
Malicious Restriction	No	Yes	No

mining server to encrypt its input (i.e., local data mining model) and exchange the encrypted input with each other. Based on mathematical manipulations, the data mining servers can build globally valid data mining models spanning their systems without disclosing private information to each other.

Since the semi-honest restriction assumes that the adversaries are incapable of deviating from the designated protocol, semi-honest-restriction-based information sharing protocols cannot be used in many practical systems where adversaries have such ability. Protocols designed based on the malicious restriction face a similar challenge, as these protocols cannot be used in systems where adversaries revise their own data to compromise the privacy of other parties. Besides, most existing protocols with the malicious restriction are computationally expensive, partially due to the fact that most of such protocols use sophisticated cryptosystems with high computational overhead (e.g., Paillier's cryptosystem [44] used in [39]).

Table 3 shows the comparison between existing work with semi-honest restriction and malicious restriction. In particular, we show which kinds of systems can, or cannot, be supported by each approach.

CHAPTER III

DATA COLLECTION PROTOCOL[*]

In this chapter, we address issues related to the design, analysis and implementation of data collection protocol, which protects privacy during the transmission of private data from the data providers to the data warehouse server. This chapter is organized as follows. We first present an overview on the data collection protocol, which introduces the design principle, reviews previous studies, and shows our contribution in this chapter. Then, we examine a common approach for data collection protocol called the randomization approach, and analyze its problems. In order to solve these problems, we introduce a new scheme for data collection protocol. The implementation of our scheme on privacy-preserving association rule mining and data classification (both are typical data mining problems extensively used in practice [29]) are presented, respectively. We evaluate the performance of our scheme both theoretically and experimentally, and discuss issues related to the implementation of our scheme in practical systems.

III.1 Overview

We first show the design principle of data collection protocol. Recall that in a privacy-preserving data mining system, the data providers and the data warehouse server share the objective of constructing accurate data mining models. Besides, the data providers have their own objective of protecting their private information. Thus, the design of data

---

[*] Part of the data reported in this chapter is reprinted with permission from "A New Scheme on Privacy-Preserving Data Classification" by Nan Zhang, Shengquan Wang, and Wei Zhao, 2005. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Pages 374-383. Copyright 2005 by ACM Press.

collection protocol aims to follow the minimum necessary rule by limiting the information obtained by the data warehouse server to the minimum amount necessary for the intended purpose of building accurate data mining models.

Previous studies recognize the fact that part of the private information is unnecessary for the construction of data mining models. It is shown that the data distribution, not the individual data values, is often sufficient for building accurate data mining models [4]. Based on this fact, the randomization approach is proposed for the data providers to add random noise to their private data before transmitting such data to the data warehouse server. The distribution of the random noise is known by both the data providers and the data warehouse server. Thereby, the data providers protect their privacy by hiding the real values of their data, while the data warehouse server can still construct accurate data mining models because it can recover the original data distribution from the randomized data based on the distribution of the random noise.

Most existing work on data collection protocol tacitly assumes that randomization is the only effective approach to protecting privacy while maintaining the accuracy of constructed data mining models[1]. As we will illustrate further in the latter part of this chapter, there are several problems with the randomization approach:

- Researchers have recently identified privacy breaches as one of the major problems with the randomization approach [38]. It is shown that the spectral

[1] There are a few exceptions, though. For example, a cryptographic approach has been proposed to strictly protect the private data [15]. Nonetheless, this approach has implementation problems that prevent it from being used in many practical situations. With this approach, a malfunctioning data provider can destroy the entire data mining results. Besides, the identities and number of data providers must be known by the data warehouse server *before* data collection begins. This is impossible in many real systems like online survey.

structure of the randomized data can help a malicious data warehouse server to recover the original data from their randomized values. Since the original data values cannot be recovered solely from the constructed data mining models, there must be a substantial amount of private information that is unnecessary for data mining but is divulged to the data warehouse server. This contradicts the minimum necessary rule and increases the risk of privacy leakage.

- Another major problem with the randomization approach is that it requires all data providers to accept the same level of privacy disclosure. Due to the results of recent surveys [14], [33], [35], [59], different people tend to have different levels of privacy concern in regards to their personal information. As such, when the randomization approach is used, the (potentially accurate) data from privacy unconcerned data providers may be wasted while privacy fundamentalists may be unwilling to disclose their data.

In this chapter, we develop a new scheme on data collection protocol based on algebraic techniques. In our scheme, the data providers do not perturb their data by only using randomly generated noise. Instead, the data warehouse server transmits *perturbation guidance* to data providers as references to the perturbation of their data. The perturbation guidance indicates which part (e.g., items, attributes) of the private data is the minimum necessary to build accurate data mining models. After validating the perturbation guidance, the data providers perturb their data accordingly. Our scheme adheres to the minimum necessary rule by transmitting only the minimum necessary

information to the data warehouse server. We will demonstrate that our new scheme has the following important features, which distinguish it from previous approaches.

- Our scheme can help build more accurate data mining models while disclosing less private information. In particular, we derive theoretical bounds on the accuracy of data mining models constructed by our scheme. Such bounds can be used to predict system accuracy in reality.
- Our scheme allows each data provider to play a role in determining the tradeoff between accuracy and privacy. Specifically, we allow each data provider to choose a different level of privacy disclosure. As such, our system can satisfy the privacy requirements of a wide range of data providers, from hard-core privacy fundamentalists to people marginally concerned about their privacy.
- Our scheme is flexible and easy to implement. For a given data mining task (e.g., classification), our scheme is transparent to the concrete data mining algorithm used (e.g., decision tree, naïve Bayesian, etc). The reason is that in our scheme, the (perturbed) data collected by the data warehouse server can be directly used as inputs to the data mining algorithms. Thus, our scheme can be readily integrated with existing systems as a middleware. Furthermore, as we will show in the latter part of this chapter, our scheme is efficient in terms of computational cost, space complexity, and communication overhead. This makes our scheme scalable to very large number of data providers.

III.2 Randomization Approach and Its Problems

We now briefly review the randomization approach, which has been a popular choice in previous studies. We will analyze the problems associated with the randomization approach, and explain reasons behind the problems, which motivate us to propose a new scheme on data collection protocol.

III.2.1 Randomization Approach

When the randomization approach is used, the data collection process consists of two steps. The first step is for the data providers to randomize their data and transmit the randomized data to the data warehouse server. As in many real systems (e.g., online survey systems) where each data provider comes at a different time, we consider this step to be iteratively carried out by a group of independent threads[2]. In each thread, a data provider applies the predetermined randomization operator on its data, and transmits the randomized data to the data warehouse server. The randomization operator is known by both the data providers and the data warehouse server.

Various randomization operators have been proposed for different kinds of data. Examples include the additive-noise operator [4] and the random response operator [19], which are shown as $R(\cdot)$ in (3.1) and (3.2), respectively,

$$R(t) = t + r. \tag{3.1}$$

$$R(t) = \begin{cases} t, & \text{if } r < \theta. \\ \overline{t}, & \text{if } r \geq \theta. \end{cases} \tag{3.2}$$

[2] Nonetheless, without loss of generality, we assume that such threads can be executed in a serializable manner.

where t is the private data tuple, r is the random noise, and θ is a predetermined parameter. In (3.2), t must be a binary data tuple and $\overline{t}$ is the logical NOT of t. There are also more complex randomization operators such as the cut-and-paste operator [24]. Most existing randomization operators satisfy the following two conditions.

- The only inputs to the randomization operator are 1) the private data, and 2) a random number generator.
- All data providers must use the same randomization operator with the same parameters. The operator and the parameters are known by both the data providers and the data warehouse server.

As the result of this step, the data warehouse server receives all randomized data from the data providers.

In the second step, the data warehouse server estimates the original distribution of the data by employing a distribution reconstruction algorithm. Several distribution reconstruction algorithms have been proposed in correspondence to different randomization operators [1], [4], [19], [24], [47]. The basic idea of most algorithms is to use Bayesian analysis to estimate the original data distribution based on the randomization operator and the randomized data. For example, the expectation maximization (EM) algorithm [1] generates a reconstructed distribution that converges to the maximum likelihood estimate of the original distribution.

Note that in the second step, a malicious data warehouse server may compromise private information of the data providers by using a private data recovery algorithm on the randomized data supplied by the data providers. This algorithm aims to recover the

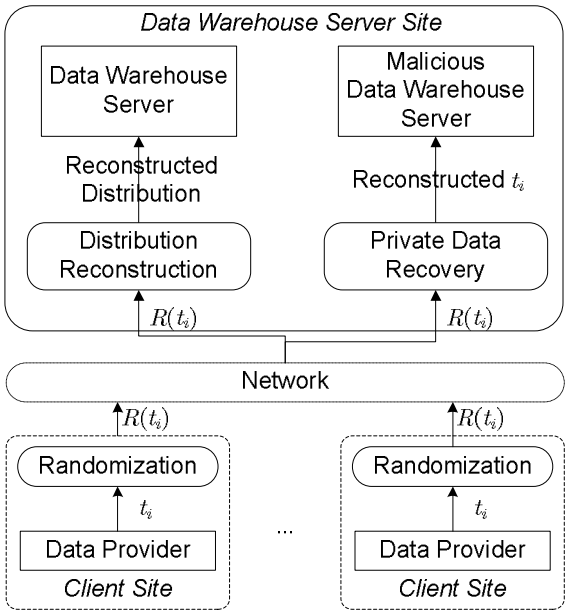

Fig. 2. Randomization Approach.

original values of the private data by separating noise from the randomized data. Several private data recovery algorithms have been proposed and discussed [1], [38]. For example, the spectral reconstruction algorithm recovers the original data values by exploiting the spectral structure of the perturbed data [38].

Figure 2 depicts the system architecture with randomization approach. The private data recovery component exists in the system only if the data warehouse server is malicious. Note that the data transmission in the system is relayed one-way: from the data providers to the data warehouse server. There is no communication from the data warehouse server to the data providers, or between different data providers.

III.2.2 Problems of the Randomization Approach

While the randomization approach is intuitive and easy to implement, it also has several problems. The most serious one is privacy breach. A private data recovery algorithm proposed in [38] can accurately recover the original data from the randomized data. The basic idea of the algorithm can be stated as follows. There is usually strong correlation between different attributes (or items) of the original data. The relationship between different attributes (or items) is well preserved after randomization. Nonetheless, the noise added to each attribute is independent. Given this fact, the algorithm uses a filtering method to exploit the spectral structure of the perturbed data, and thereby separates noise from the perturbed data.

Another problem with the randomization approach is that it cannot satisfy the heterogeneous privacy requirements of data providers, which have been highlighted by multiple survey results. For example, survey results on the U.S. public's attitudes toward privacy protection [59] indicate that different people tend to have different levels of privacy concern in regard to their personal information: There are 37% privacy fundamentalists, who are extremely concerned about their privacy, 52% privacy pragmatists, who believe there should be a tradeoff between privacy and public interest, and 11% who are generally unconcerned about their privacy. Similar results have been reported by surveys conducted in U.K. [33], Germany [33], and Japan [35]. Such divergence on privacy concern can also be observed from the wide coverage of public debates between privacy and freedom of expression [43].

Randomization approach treats all data providers in the same manner and does not address the varying privacy concerns of the data providers. In many cases, a privacy fundamentalist may not be willing to send out its data while the (potentially accurate) data from privacy unconcerned people are wasted.

Randomization approach also suffers from efficiency problem, as it places a heavy load (i.e., distribution reconstruction) on the data warehouse server at the critical time path of the data mining process. The distribution reconstruction is at the critical time path because it must take place after all data have been collected, and before these data can be used for data mining. It has been shown that the computational cost of constructing data mining models on randomized data can be "orders of magnitudes more than" that of constructing data mining models on the original data [7].

III.2.3 Reasons behind the Problems of Randomization Approach

We believe that below are the reasons behind the problems of randomization approach.

- The cause of privacy breach problem is that the randomization approach is attribute-invariant. That is, every attribute (or item), no matter how useful for data mining, is equally perturbed in the randomization process. Apparently, there may be a substantial amount of private information unnecessary (or less necessary) for data mining but is divulged to the data warehouse server. Since all attributes are equally perturbed, the relationship between different attributes of the original data is well preserved after the randomization. Thus, the spectral structure of the original data is also well preserved after the randomization. Consequently, the spectral structure of the perturbed data can be used to recover

the original data values. This problem can be solved by allowing the data providers to submit only those attributes that are most necessary for data mining. Nonetheless, this solution is infeasible with the randomization approach because the communication flows one-way: from the data providers to the data warehouse server. As such, a data provider cannot learn which attributes are more important for data mining, and therefore has no choice but to perturb its data in an attribute-invariant manner.

- The reason why the randomization approach cannot satisfy different privacy requirements of data providers is because the randomization approach is user-invariant. That is, all data providers share the same randomization operator, and thus must accept the same (expected) level of privacy disclosure. This problem can be solved by allowing each data provider to negotiate its own level of privacy disclosure with the data warehouse server. However, this solution is infeasible with the randomization approach because, again, there is no communication from the data warehouse server to the data providers. As such, a data provider cannot receive any user-specific guidance on the perturbation of its data, and has no choice but to perturb its data in a user-invariant manner.
- An important reason behind the efficiency problem is that the distribution reconstruction algorithm is not incremental. Recall that data collection is considered to be an iterative process as each data provider submits its data at a different time. If the distribution reconstruction algorithm could be deployed incrementally, the data warehouse server would be able to estimate the original

distribution when the data are being collected, and to update the estimated distribution when a new (perturbed) data tuple arrives. As such, the distribution reconstruction process would no longer be at the critical time path of privacy-preserving data mining, and thus the efficiency problem could be solved.

III.3 Our New Scheme

The one-way communication scheme and the non-incremental nature of distribution reconstruction are inherent in the design of randomization approach. This motivates us to propose a new scheme on data collection protocol. Compared with the randomization approach, our new scheme allows two-way communication between the data warehouse server and the data providers. Besides, our scheme does not require any post-collection processing of the collected data (i.e., after the data are collected and before the data are used in data mining).

Another possible method for two-way communication is to introduce data transmission between data providers. We do not adopt this method because in many practical systems (e.g., online survey systems), different data providers may come at different time and thus may not be able to communicate with each other. Other potential weaknesses of this method include: 1) it requires the trustworthiness of other data providers, and 2) it may place considerable computational load on the data providers, which have considerably lower computational power than the data warehouse server.

III.3.1 Basic Idea

The basic idea of our scheme is for the data providers to submit only the part of their private information that is most necessary for constructing accurate data mining models.

In order to for the data providers to know which part of the private data is necessary for data mining, we introduce two-way communication between the data warehouse server (which has a global view of received data) and the data providers.

With our scheme, before perturbing its data, a data provider first receives *perturbation guidance* from the data warehouse server. The perturbation guidance is computed by the data warehouse server based on the (perturbed) data that it has already received. Note that the original data are usually high-dimensional as the data contain multiple attributes or items. Generally speaking, the perturbation guidance represents a dimension-reduction function that projects the originally high-dimensional private data into a low-dimensional subspace such that the private information retained in the projected data is the most necessary for data mining.

After validating the received perturbation guidance, the data providers perturb their data accordingly. The data warehouse server can directly store the perturbed data in the data warehouse, and use such data to support data mining. Thus, our scheme does not need any processing (e.g. distribution reconstruction) between the collection of data and the construction of data mining models.

III.3.2 System Architecture

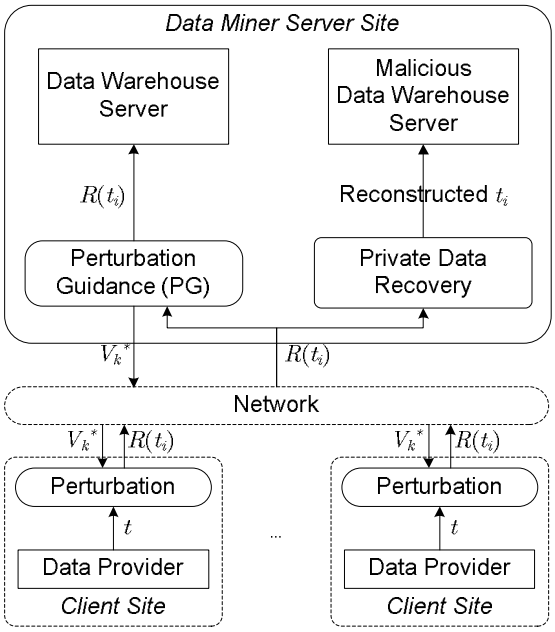

Fig. 3. Our New Scheme.

Figure 3 depicts the system architecture with our new scheme. In our scheme, we assume that the data warehouse server may be malicious. That is, the data warehouse server may do whatever it can in order to compromise the private information of the data providers.

The communication protocol of our scheme is shown in Table 4. In our scheme, there is an important parameter for each data provider, called the *maximum acceptable privacy disclosure level*, which is denoted by k_i (i is the index of the data provider). Roughly speaking, if we consider the private data tuple as a random vector, then k_i is the degree of freedom [50] of the perturbed data tuple, which in most cases is much smaller than the degree of freedom of the original data tuple. Generally speaking, k_i can also be understood as the dimensionality of the space formed by the perturbed data tuples. The larger k_i is, the more contribution the perturbed data tuple will make to the construction of data mining models. Nonetheless, with a larger k_i, the data warehouse server will also

TABLE 4

Communication Protocol

For the data warehouse server:
S1. Upon receiving an inquiry message from a data provider, the perturbation guidance (PG) component computes the current system disclosure level k^* and sends it to the data provider;
S2. Upon receiving a ready message from a data provider, the PG component computes V_k^* and sends it to the data provider;
S3. Upon receiving a perturbed data tuple $R(t_i)$ from a data provider, sends $R(t_i)$ to the PG component.
For the data providers:
Input: k_i, the maximum acceptable disclosure level of the data provider.
C1. Wait until the data is ready;
C2. Sends an inquiry message to the data warehouse server to obtain the current system disclosure level k^*;

C3. if the received k^* is less than or equal to k_i then
Sends a ready message to the data warehouse server.
else
Wait for a random amount of time; Goto C1;
end if

C4. Upon receiving V_k^* from the data warehouse server, sends V_k^* to the perturbation component to check its validity. If V_k^* is valid, the perturbation component perturbs the private data tuple based on V_k^* and sends the perturbed data tuple to the data warehouse server.

have more information about individual private data. Therefore, a privacy fundamentalist can choose a small k_i to protect its privacy. A privacy-unconcerned individual can choose a large k_i to help build more accurate data mining models.

Before requesting for the current perturbation guidance, a data provider first inquires the data warehouse server for the current system disclosure level k^*, which is the disclosure level needed for the data warehouse server to construct accurate data mining models. The perturbation guidance component of the data warehouse server computes k^* and transmits it to the data provider. If k^* is unacceptable by the data provider (i.e., $k^* > k_i$), the data provider can wait for a period of time and try again. As we will show in the experimental evaluation, k^* decreases rapidly when the number of data tuples received by the data warehouse server increases. Since the level of privacy concern varies among different data providers [59], the system disclosure level will be acceptable by most data providers in a short period of time.

If k^* is accepted by a data provider (i.e., $k^* \leq k_i$), the data provider inquires for the current system perturbation guidance, denoted by V_k^*. The data warehouse server computes V_k^* and dispatches it to the data providers. The perturbation guidance V_k^* is supposed to project the original data tuple into a k^*-dimensional subspace such that the private information retained in the projected data is the most necessary information for data mining. Note that as we will show in the latter part of the chapter, the computation of V_k^* is determined by specific data mining tasks.

Once V_k^* is received and validated, the data provider computes its perturbed data tuple $R(t_i)$ based on the original data tuple t_i and the perturbation guidance V_k^*, and then transmits $R(t_i)$ to the data warehouse server. After the data warehouse server receives all perturbed data tuples, the data warehouse server directly stores the received data in the data warehouse and uses such data to support data mining.

As we can see from Table 4, our scheme requires two rounds of message exchange to dispatch the perturbation guidance: one round to inquire k_i and another round to inquire V_k^*. One may consider an alternative approach by asking each data provider to transmit its maximum acceptable privacy disclosure level k_i to the data warehouse server. If there is $k^* \leq k_i$, the data warehouse server transmits V_k^* back to the data provider. This approach only requires one round of message exchange. However, privacy breaches may occur when this approach is used because when $k^* < k_i$, a malicious data warehouse server can manipulate a disclosure level $\hat{k}$ with $k^* < \hat{k} < k_i$, and then generate a perturbation guidance based on $\hat{k}$. As such, the data warehouse server may compromise private data that are unnecessary for data mining.

Compared with the randomization approach, our scheme does not have the distribution reconstruction component. Instead, the collected (perturbed) data can be directly stored in the data warehouse and used to support data mining. Our scheme has two key components, which are the perturbation guidance (PG) component of the data warehouse server, and the perturbation component of the data providers. The design of the perturbation guidance components depends on specific data mining tasks (e.g.,

association rule mining, data classification). Although the design of the perturbation component is independent of data mining tasks, the description of it requires notions introduced with the perturbation guidance component. Thus, we will introduce these two components for association rule mining and data classification, respectively, as follows.

III.4 Design for Association Rule Mining

We now implement our scheme to support privacy-preserving mining of association rules. Recall that there are two basic components in our scheme: 1) the perturbation guidance component of the data warehouse server, which computes the current system privacy disclosure level k^* and the perturbation guidance V_k^*, and 2) the perturbation component of the data providers, which validates V_k^* and perturbs the private data. We will present the implementation of these components for privacy-preserving association rule mining systems after introducing notions of the private dataset.

III.4.1 Basic Notions

Let there be m data providers in the system, each of which holds a private transaction t_i ($i \in [1, m]$). Let I be a set of n items $a_1, \ldots, a_n$. Each transaction t_i is a set of items such that $t_i \subseteq I$. The data warehouse server has no external knowledge about the private information of data providers.

We represent each transaction by an n-dimensional binary vector t_i such that the j-th bit of the vector is 1 if and only if $a_j \in t_i$. Correspondingly, we represent the set of all

private data tuples by an $m \times n$ matrix $T = [t_1; \ldots; t_m]$.[3] Each transaction t_i is represented by a row vector in T. We denote the transpose of T by T'. We use $\langle T \rangle_{ij}$ to denote the element of T with indices i and j.

Based on these notions, we briefly review the definition of association rule mining. More detailed description of association rule mining can be found in classic textbooks (e.g., [29]). The major objective of association rule mining is to identify all frequent itemsets (i.e., sets of items). An itemset B is *frequent* if and only if its support $supp(B)$ is larger than or equal to a predetermined minimum threshold *min_supp*. The support of B is defined as follows.

$$supp(B) = \frac{\#\{t \in T \mid B \subseteq t\}}{m}. \tag{3.3}$$

That is, the support of B is the percentage of transactions in T that contain B as a subset.

III.4.2 Perturbation Guidance Component

As we are considering the case where perturbed transactions are iteratively fed to the data warehouse server, the data warehouse server maintains a set of all received transactions and updates the set when a new transaction arrives. Let the current matrix of received transactions be T^*. When a new (perturbed) transaction $R(t_i)$ arrives, $R(t_i)$ is appended to T^*. Without loss of generality, we assume that $R(t_i)$ is the i-th transaction received by the data warehouse server. As such, when the data warehouse server receives m^* transactions, T^* is an $m^* \times n$ matrix $[R(t_1); \ldots; R(t_{m^*})]$.

---

[3] In the context of private dataset, t_i is a transaction. In the context of matrix, t_i is the corresponding row vector in T.

In order to compute the current system disclosure level and the perturbation guidance for the first data provider, we assume that before the data collection process begins, the data warehouse server already has m_0 ($m_0 << m$) randomly generated transactions in T^*.

Besides the matrix of received transactions T^*, the data warehouse server also keeps track of an $n \times n$ symmetric matrix $A^* = T^{*\prime} T^*$. Note that

$$A^* = \sum_{i=1}^{m^*} R(t_i)' R(t_i). \tag{3.4}$$

Thus, the update of A^* (after $R(t_{m^*})$ is received) does not need access to any transaction other than the recently received $R(t_{m^*})$. As such, we do not require matrix T^* to remain in the main memory during data collection.

We now show how to compute the current system privacy disclosure level k^* and the perturbation guidance V_k^* based on A^*. Since A^* is a symmetric matrix, we can use eigen decomposition to decompose A^* as

$$A^* = V^* \Sigma^* V^{*\prime}, \tag{3.5}$$

where $\Sigma^* = \text{diag}(\sigma_1^*, \cdots, \sigma_n^*)$ is a diagonal matrix with diagonal elements $\sigma_1^* \leq \cdots \leq \sigma_n^*$, σ_i^* is the i-th eigenvalue of A^*, and V^* is an $n \times n$ unitary matrix composed of the eigenvectors of A^* (as the column vectors). As we will show in the theoretical analysis, an appropriate choice of the system privacy disclosure level k^* should be the minimum degree of freedom of A^* that maintains an accurate estimate of $A = T' T$. Based on the eigen decomposition of A^*, we can compute k^* as the minimum integer that satisfies

$$\sigma^*_{k^*+1} \leq \mu\sigma^*_1, \tag{3.6}$$

where μ is a parameter predetermined by the data warehouse server. A data warehouse server that desires high accuracy of data mining results can choose a small μ to ensure an accurate estimation of A. A data warehouse server that can tolerate a relatively lower level of accuracy can choose a large μ to help protect data providers' privacy. In order to choose a good cutoff k^* to retain sufficient information about A, a textbook heuristic is to set $\mu = 15\%$ [28].

Given k^*, the perturbation guidance V_k^* is an $n \times k^*$ matrix that is composed of the first k^* eigenvectors of A^* (i.e., the first k^* column vectors of V^* corresponding to the k^* largest eigenvalues of A^*). In particular, if $V^* = [v_1; \ldots; v_n]$, then $V_k^* = [v_1; \ldots; v_{k^*}]$. Since V^* is a unitary matrix, there is $V_k^{*'} V_k^* = I$, where I is the $k^* \times k^*$ identity matrix.

We note that due to efficiency and privacy concerns, the data warehouse server only updates k^* and V_k^* once several data tuples are received. The efficiency concern is the overhead of computing k^* and V_k^*. The privacy concern is that if V_k^* is updated once every data tuple is received, a malicious data provider may infer the newly submitted (perturbed) data tuple by tracking the change of V_k^*. Although a data provider is comfortable transmitting the perturbed data tuple to the data warehouse server, it may be unwilling to divulge it to another data provider. The justification of k^* and V_k^* will be provided in the theoretical analysis of our scheme. The runtime efficiency and the communication overhead will be discussed in the implementation issues.

III.4.3 Perturbation Component

Recall that our implementation of perturbation component is independent of data mining tasks. Thus, the perturbation component presented here can also be used for privacy-preserving data classification.

The perturbation component has two objectives: 1) to validate V_k^*, and 2) to perturb the private data t_i based on V_k^*. The perturbation component validates V_k^* by checking if V_k^* is an $n \times k^*$ matrix that satisfies $V_k^{*\prime} V_k^* = I$, where I is the $k^* \times k^*$ identity matrix. If V_k^* is valid, the perturbation component perturbs t_i in a two-step process. Recall that t_i is represented by an n-dimensional row vector with elements of either 0 or 1. In the first step, t_i is perturbed to be another n-dimensional row vector $\tilde{t}_i$, such that

$$\tilde{t}_i = t_i V_k^* V_k^{*\prime}. \tag{3.7}$$

Note that the elements of $\tilde{t}_i$ may be real values. Thus, we need a second step to transform $\tilde{t}_i$ to $R(t_i)$ such that each element in $R(t_i)$ is either 0 or 1. In the second step, for all $j \in [1, n]$, the data provider generates a random number r_j that is chosen uniformly at random from [0, 1], and computes $R(t_i)$ as follows.

$$\langle R(t_i) \rangle_j = \begin{cases} 1, & \text{if } r_j \le \langle \tilde{t}_i \rangle_j^2, \\ 0, & \text{otherwise,} \end{cases} \tag{3.8}$$

where $\langle \cdot \rangle_j$ is the j–th element of a vector. As we can see, the probability that $\langle R(t_i) \rangle_j = 1$ is equal to $\langle \tilde{t}_i \rangle_j^2$.

III.5 Design for Data Classification

We now apply our scheme on privacy-preserving data classification problem.

III.5.1 Basic Notions

Most of the notions are similar to those proposed in association rule mining. Thus, we only introduce those notions that are new or have different meanings. Let there be m data providers in the system, each of which holds a private data tuple t_i ($1 \leq i \leq m$) and its class label a_0. The private data tuple consists of n attributes a_1, …, a_n. The class label attribute is insensitive and indicates which predefined class the data tuple belongs to. All other attributes are private information and need to be protected. Without loss of generality, we assume that the data warehouse server knows the class label of each data tuple but has no external knowledge about the private information of data providers. The main purpose of data classification is to construct a model (i.e., classifier) to predict the class labels of testing data tuples based on the training dataset (i.e., data providers' data), where the class label of each data tuple is given. More detailed description of data classification can be found in classic textbooks (e.g., [29]).

In this study, we assume that there are two classes C_0 and C_1. The class label attribute has two distinct values 0 and 1, corresponding to classes C_0 and C_1, respectively. We first consider the cases where all attributes are categorical (i.e., discrete-valued). If the value of an attribute is continuous, the attribute can be discretized first. An example of such discretion is provided in the experimental evaluation of our scheme. Let the number of distinct values of a_j be s_j. Without loss of generality, let the value range of a_j

be 0, …, $s_j - 1$. We denote a private data tuple t_i by an $(s_1 + \cdots + s_n)$-dimensional binary vector as follows.

$$t_i = \overbrace{0,...,1}^{s_1 \text{ bits for } a_1},...,\overbrace{0,...,1}^{s_n \text{ bits for } a_n},...,0. \tag{3.9}$$

Within the s_j bits for a_j, the h-th bit is 1 if and only if $a_j = h - 1$.

Although our scheme applies to all categorical attributes (with arbitrary s_j), for the simplicity of discussion, we assume that all attributes a_1, …, a_n are binary. That is, $s_1 = \cdots = s_n = 2$. As such, each private data tuple can be represented by a $2n$-dimensional binary vector. Accordingly, we denote the private training dataset by an $m \times 2n$ matrix $T = [t_1; \ldots; t_m]$. Let T_0 and T_1 be the matrices that represent the private data tuples in class C_0 and C_1, respectively. T_0^* and T_1^* are defined in analogy to the definition of T^*. We denote the number of data tuples in T_i by $|T_i|$. An example of T is shown in Table 5. As we can see from the example, data tuple t_1 belongs to class C_1 and has three attributes $[a_1, a_2, a_3] = [1, 0, 0]$.

TABLE 5

Example of Matrix T

	a_0
t_1	1
.	.
.	.
.	.
t_m	0

Class label:

T:

	a_1	a_2	a_3
t_1	0,1	1,0	1,0
.	.	.	.
.	.	.	.
.	.	.	.
t_m	1,0	0,1	1,0

III.5.2 Perturbation Guidance Component

The perturbation guidance component for privacy-preserving data classification is similar to that for privacy-preserving mining of association rule. The only difference (besides the difference on the size of matrices) is the computation of A^*. In the perturbation guidance component for privacy-preserving data classification, the data warehouse server keeps track of two matrices $A_0^* = T_0^{*\prime} T_0^*$ and $A_1^* = T_1^{*\prime} T_1^*$, and computes A^* as $A^* = A_0^* - A_1^*$. Other variables, including V^*, Σ^*, the current system privacy disclosure level k^*, and the perturbation guidance V_k^*, are still computed in the same manner based on the eigen decomposition of A^*.

III.6 Performance Analysis

We now analyze the performance of our new scheme. Clearly, a privacy-preserving data mining approach should be measured by its capability of both protecting private information and constructing accurate data mining models. Thus, we measure on 1) the amount of privacy disclosure, and 2) the accuracy of constructed data mining models. In particular, the privacy measure is independent of specific data mining tasks while the accuracy measure depends on these tasks. We will derive theoretical bounds on the measures, in order to provide guidelines for the tradeoff between privacy and accuracy, and thereby help system managers set parameters in practice.

III.6.1 Privacy Analysis

We first define our privacy measure. Then, we derive an upper bound on the amount of privacy disclosure based on a data provider's maximum acceptable privacy disclosure level.

In our scheme, we need to guarantee that for any private data tuple (or transaction) t, the data warehouse server cannot deduce the value of t from the perturbed data tuple $R(t)$. In particular, since the data warehouse server can be malicious, we must consider the cases where the data warehouse server manipulates the system perturbation level k^* and/or the perturbation guidance V_k^* to compromise the private information of data providers.

Recall that our scheme allows each data provider to choose a different privacy disclosure level. Thus, we define the privacy disclosure measure for individual data providers. Formally, let the maximum acceptable disclosure level of a data provider be k_i. Let the private data tuple of the data provider be t_i. For any matrix $\hat{V}_k^*$ that can pass the validation test, let $R(t,\hat{V}_k^*)$ be the output of $R(t_i)$ when the received perturbation guidance is $\hat{V}_k^*$. With these notions, we define the privacy measure as follows.

Definition 3.1. Suppose that the data warehouse server sends an arbitrary integer k^* as the current system disclosure level and an arbitrary matrix $\hat{V}_k^*$ as the perturbation guidance. The degree of privacy disclosure $l_p(k_i)$ is defined by the maximum percentage of private information disclosed by the perturbed data tuple $R(t,\hat{V}_k^*)$. That is,

$$l_{\mathrm{p}}(k_i) = \max_{\hat{V}_k^* | k^* \le k_i} \left(\frac{I(t; R(t, \hat{V}_k^*))}{H(t)} \right) \tag{3.10}$$

$$= \max_{\hat{V}_k^* | k^* \le k_i} \left(1 - \frac{H(t \mid R(t, \hat{V}_k^*))}{H(t)} \right) \tag{3.11}$$

where $I(t; R(t, \hat{V}_k^*))$ is the mutual information between t and $R(t, \hat{V}_k^*)$, and $H(\cdot)$ denotes the information entropy.

In the definition, the mutual information $I(t; R(t, \hat{V}_k^*))$ measures the amount of private information about t that remains in $R(t, \hat{V}_k^*)$ [13]. The information entropy $H(t)$ measures the amount of information in t [13]. As we can see, the degree of privacy disclosure measures the maximum percentage of private information divulged to the data warehouse server.

There are two kinds of privacy measures that have been proposed in the literature: information theoretic measure [1], which we use in the above definition, and privacy breach measure [23]. We now briefly compare these types of measures and explain the reason why we adopt the information theoretic measure.

As illustrated by our definition, the information theoretic measure uses mutual information to measure the *average* amount of private disclosure when the original data take all possible values. The privacy breach measure, instead, measures the amount of privacy disclosure in the worst case. For example, suppose that a binary attribute a has an extremely small (priori) probability to be 1. Nonetheless, if $a = 1$, there is a considerably high (posterior) probability that the data warehouse server can infer $a = 1$

from the perturbed value of a. Suppose that when $a \neq 1$, the amount of privacy disclosure is fairly small. The privacy breach measure considers such perturbation scheme as ineffective because in the worst case (when $a = 1$), the probability of privacy breach is high. On the other hand, the information theoretic measure considers such perturbation scheme as effective because the overall expected amount of privacy disclosure is still fairly small.

Both information theoretic measure and privacy breach measure have their applicable domains. The information theoretic measure can be used in scenarios where the average amount of privacy disclosure is of concern. The privacy breach measure can be used in scenarios where the collected data are more sensitive and thus require more rigid guarantee on privacy protection, even in the worst case.

We use the information theoretic measure to measure the performance of our scheme because from privacy protection perspective, our scheme "discriminates against" certain attributes that are most necessary for data mining. For example, as we will show in the experimental evaluation, our scheme discloses little information about attributes unnecessary for data mining, but discloses much more information about attributes critical for the accuracy of constructed data mining models. If we use the privacy breach measure to gauge the worst-case privacy disclosure, the result will be biased due to the relatively high, albeit necessary, level of privacy disclosure on attributes most necessary for data mining.

Based on Definition 3.1, we now derive an upper bound on the degree of privacy disclosure when our scheme is used.

Theorem 3.1. With our scheme, we have

$$l_{\mathrm{p}}(k_i) \leq \frac{\sigma_1 + \cdots + \sigma_{k_i}}{mn}, \tag{3.12}$$

where σ_i is the j–th eigenvalue of $A = T' T$.

Proof: Recall that each original data tuple (or transaction) is represented by a row vector in T. Since the size of T varies by data mining tasks, without loss of generality, we assume that T is an $m \times n_{\mathrm{s}}$ matrix ($m \gg n_{\mathrm{s}}$). Consider matrix $T \hat{V}_k^* \hat{V}_k^{*\prime}$, each row vector of which is $\tilde{t}_i = t_i \hat{V}_k^* \hat{V}_k^{*\prime}$. Since $\hat{V}_k^*$ has to be an $n_{\mathrm{s}} \times k^*$ matrix in order to pass the validation test, the rank of $\hat{V}_k^*$ is no more than k^*. Thus, the rank of $T\hat{V}_k^* \hat{V}_k^{*\prime}$ is less than or equal to k^*. Due to the properties of low-rank matrix approximation [28], we have

$$\left\| T - T\hat{V}_k^* \hat{V}_k^{*\prime} \right\|_F \geq \sqrt{\sigma_{k^*+1} + \cdots + \sigma_{n_{\mathrm{s}}}}, \tag{3.13}$$

where $||\cdot||_F$ is the Frobenius norm of a matrix (i.e., the square root of the sum of the squares of all elements in the matrix). Also note that since $\hat{V}_k^*$ passes the validation test, we have $\hat{V}_k^{*\prime} \hat{V}_k^* = I$. Given the computation of $R(t_i)$ based on $\tilde{t}_i$, we have

$$\underset{R(t_i)}{\mathrm{Exp}}((\langle R(t_i) \rangle_j - \langle t_i \rangle_j)^2) = \begin{cases} \langle \tilde{t}_i \rangle_j^2 = (\langle \tilde{t}_i \rangle_j - \langle t_i \rangle_j)^2, & \text{if } \langle t_i \rangle_j = 0, \\ 1 - \langle \tilde{t}_i \rangle_j^2 > (\langle \tilde{t}_i \rangle_j - \langle t_i \rangle_j)^2, & \text{if } \langle t_i \rangle_j = 1, \end{cases} \tag{3.14}$$

where Exp(·) denotes the expected value. Let T_{R} be the matrix of perturbed data tuples. Each perturbed data tuple is a row vector in T_{R}. We have

$$\left\| T - T_{\mathrm{R}} \right\|_F \geq \left\| T - T\hat{V}_k^* \hat{V}_k^{*\prime} \right\|_F \geq \sqrt{\sigma_{k^*+1} + \cdots + \sigma_{n_s}}, \tag{3.15}$$

Consider the transformation of an element from $\langle t_i \rangle_j$ to $\langle R(t_i) \rangle_j$. Due to our computation of $\langle R(t_i) \rangle_j$, almost all transformations are from 1 to 0. Thus, the number of elements in T_{R} that are equal to 1 is less than $\sigma_{k^*+1} + \cdots + \sigma_{n_s}$. As such, for any attribute (or item) a_j in t_i, the probability that all the elements in $R(t_i)$ representing a_j are 0 is greater or equal to than $1 - (\sigma_{k^*+1} + \cdots + \sigma_{n_s})/mn$. Following the definition of mutual information, we have

$$I(t; R(t, \hat{V}_k^*)) \geq \frac{\sigma_{k^*+1} + \cdots + \sigma_{n_s}}{mn} H(t) \tag{3.16}$$

Since $k^* \leq k_i$, the degree of privacy disclosure satisfies

$$l_{\mathrm{p}}(k_i) \leq \frac{\sigma_1 + \cdots + \sigma_{k_i}}{mn}. \tag{3.17}$$

□

As we can see from the theorem, the less k_i is, the less the upper bound will be. Thus, a data provider can always control the amount of private disclosure by adjusting its maximum acceptable privacy disclosure level k_i.

III.6.2 Accuracy Analysis

We first analyze the accuracy of our scheme on mining association rules. As we mentioned in the design of our scheme, we focus on the task of identifying frequent itemsets with support larger than a predetermined threshold *min_supp*. Thus, we use the error on the support of itemsets to measure the accuracy of our scheme.

Recall that I is the set of all items. Given itemset $B \subseteq I$, let $supp(B)$ and $supp_P(B)$ be the support of B in the original dataset and the perturbed dataset, respectively. We define the accuracy measure as follows.

Definition 3.2. The degree of error l_e for association rule mining is the maximum error on the support of an itemset. That is,

$$l_e = \max_{B \subseteq I} | supp(B) - supp_P(B) |. \tag{3.18}$$

Generally speaking, the degree of error measures the discrepancy between the frequent itemsets mined from the original dataset and those mined from the perturbed dataset. Recall that μ is a predetermined parameter used by the data warehouse server to compute the current system perturbation level k^*. Also recall that $A = T^{'}T$. We now derive an upper bound on the degree of error as follows.

Theorem 3.2. With our scheme, when m is sufficiently large, we have

$$l_e \leq \frac{2\mu\sigma_1}{m}, \tag{3.19}$$

where σ_1 is the largest eigenvalue of A.

Proof: We first prove that in the cases where $|B| \leq 2$ (i.e., B contains 1 or 2 items), there is $\max_B |supp(B) - supp_P(B)| \leq 2\mu\sigma_1/m$. Then, we extend our proof to the cases where $|B| > 2$.

Before presenting our proof, we first show an intuitive explanation of matrix A. Consider an element of A with indices i and j, respectively, we have

$$\langle A \rangle_{i,j} = \sum_t \langle t \rangle_i \langle t \rangle_j = \#\{t \mid \{a_i, a_j\} \subseteq t\}, \tag{3.20}$$

which is the number of original transactions that contain both a_i and a_j. Note that for all a_i and a_j, there is

$$supp(\{a_i, a_j\}) = \frac{\#\{t \mid \{a_i, a_j\} \subseteq t\}}{m}. \tag{3.21}$$

Thus, we have

$$\langle A \rangle_{i,j} = m \cdot supp(\{a_i, a_j\}). \tag{3.22}$$

As we can see, when $|B| = 1$, $\max_B|supp(B) - supp_P(B)|$ is in proportion to the maximum error on the estimate of the diagonal elements of A. When $|B| = 2$, $\max_B|supp(B) - supp_P(B)|$ is in proportion to the maximum error on the estimate of non-diagonal elements of A. Let the matrix of perturbed dataset be T_R. Let the estimation of A derived from T_R be A_R (i.e., $A_R = T_R' T_R$). We now derive an upper bound on $\max_{ij}|\langle A - A_R \rangle_{ij}|$.

Recall that in the first step of the perturbation, a data provider computes $\tilde{t}_i = t_i V_k^* V_k^{*\prime}$. Let $\tilde{T}$ be the $m \times n$ matrix composed of all $\tilde{t}_i$ (i.e., $\tilde{T} = [\tilde{t}_1; \ldots; \tilde{t}_m]$). In our scheme, V_k^* contains the first k^* eigenvectors of the current A^*. When m is sufficiently large, V_k^* can be approximated as the first k^* eigenvectors of A. In real cases, as m^* increases, the first k^* eigenvectors of A^* converge to those of A fairly quickly.

Let Σ_k^* be a $k^* \times k^*$ diagonal matrix with the diagonal elements be the first k^* eigenvalues of A (i.e., $\Sigma_k^* = \text{diag}(\sigma_1, \ldots, \sigma_{k^*})$). We have

$$\tilde{A} = \tilde{T}'\tilde{T} = V_k^* V_k^{*\prime} T' T V_k^* V_k^{*\prime} = V_k^* \Sigma_k^* V_k^{*\prime}. \tag{3.23}$$

That is, $\tilde{A}$ is the k^*-truncation of A [28]. As such, $\tilde{A}$ is the optimal rank-k^* approximation of A in the sense that within all rank-k^* matrices, $\tilde{A}$ has the minimum $\left\|A - \tilde{A}\right\|_2$. In particular, we have

$$\left\|A - \tilde{A}\right\|_2 = \sigma_{k^*+1}. \tag{3.24}$$

As we can see from the determination of system disclosure level k^*, our scheme maintains a cutoff k^* such that $\sigma_{k^*+1} \le \mu\sigma_1$. Thus, we have

$$\left\|A - \tilde{A}\right\|_2 \le \mu\sigma_1. \tag{3.25}$$

Since the absolute value of each element in an matrix is no larger than the 2-norm of the matrix [28], we have

$$\max\left|\left\langle A - \tilde{A}\right\rangle_{ij}\right| \le \left\|A - \tilde{A}\right\|_2 \le \mu\sigma_1. \tag{3.26}$$

As we can see from the computation of $R(t)$, for all $i, j \in [1, n]$, we have

$$\left\langle \tilde{t}\right\rangle_i^2 = \mathrm{Exp}\left(\left\langle R(t)\right\rangle_i^2\right), \tag{3.27}$$

and

$$\mathrm{Exp}\left(\left\langle t\right\rangle_i \left\langle t\right\rangle_j - \left\langle R(t)\right\rangle_i \left\langle R(t)\right\rangle_j\right) \le 2\left(\left\langle t\right\rangle_i \left\langle t\right\rangle_j - \left\langle \tilde{t}\right\rangle_i \left\langle \tilde{t}\right\rangle_j\right), \tag{3.28}$$

where Exp(·) denotes the expected value. Thus, when m is sufficiently large, we have $\max_B|supp(B) - supp_{\mathrm{P}}(B)| \le 2\mu\sigma_1/m$ for $|B| \le 2$.

We now extend the bound to the cases where $|B| \geq 3$. Let there be $|B| = h$. Without loss of generality, we show that $\left|supp(\{a_1,\ldots,a_h\}) - supp_{\mathrm{P}}(\{a_1,\ldots,a_h\})\right| \leq 2\mu\sigma_1 / m$. Note that the 2-norm of $A - \tilde{A}$ satisfies

$$\left\|A-\tilde{A}\right\|_2 = \max_{\|x\|_2=1} \left\|\left(A-\tilde{A}\right)x\right\|_2, \tag{3.29}$$

where x is an n-dimensional vector. Let

$$x = [\overbrace{\sqrt{h}/h, \ldots, \sqrt{h}/h}^{h \text{ items of } \sqrt{h}/h}, \overbrace{0, \ldots, 0}^{n-h \text{ items of } 0}]. \tag{3.30}$$

It is easy to check that $\|x\|_2 = \sqrt{h \cdot (1/h)} = 1$. As we can see,

$$\left\|\left(A-\tilde{A}\right)x\right\|_2^2 \geq \frac{1}{h}\sum_{i=1}^{h}\left(err(a_i) + \sum_{j=1, j\neq i}^{h} err(a_i, a_j)\right)^2, \tag{3.31}$$

where $err(a_i) = \left\langle A-\tilde{A}\right\rangle_{ii}$ and $err(a_i, a_j) = \left\langle A-\tilde{A}\right\rangle_{ij}$. It is easy to show that the following inequalities hold:

$$\left|\sum_t \prod_{i=1}^{h} \langle t\rangle_i - \sum_t \prod_{i=1}^{h} \langle \tilde{t}\rangle_i\right| \leq \sum_{i=1}^{h} |err(a_i)|, \tag{3.32}$$

and

$$\left|\sum_t \prod_{i=1}^{h} \langle t\rangle_i - \sum_t \prod_{i=1}^{h} \langle \tilde{t}\rangle_i\right| \leq \sum_{i=1}^{h} \sum_{j=1, j\neq i}^{h} \left|err(a_i, a_j)\right|. \tag{3.33}$$

From (3.31), (3.32), and (3.33), we have

$$\left|\sum_{t}\prod_{i=1}^{h}\langle t\rangle_i-\sum_{t}\prod_{i=1}^{h}\langle \tilde{t}\rangle_i\right|\le\left\|\left(A-\tilde{A}\right)x\right\|_2\le\mu\sigma_1. \tag{3.34}$$

Also note that (3.28) can be generalized as follows.

$$\mathrm{Exp}\left(\left|\sum_{t}\prod_{i=1}^{h}\langle t\rangle_i-\sum_{t}\prod_{i=1}^{h}\langle R(t)\rangle_i\right|\right)\le 2\left(\left|\sum_{t}\prod_{i=1}^{h}\langle t\rangle_i-\sum_{t}\prod_{i=1}^{h}\langle \tilde{t}\rangle_i\right|\right). \tag{3.35}$$

Similar to the proof of the cases where $|B| \le 2$, we have $\max_B|supp(B) - supp_P(B)| \le 2\mu\sigma_1/m$ for $|B| \ge 3$. □

Note that the expected value of σ_1 is in proportion to m because the expected value of each element in A is in proportion to m. As we can see from the theorem, the upper bound on the degree of error is proportional to, and is (almost) solely controlled by, the parameter μ. Thus, the data warehouse server can always control the accuracy of association rule mining by adjusting μ, even when the number of data providers is unknown before the data are actually collected (e.g., in online survey).

We now analyze the accuracy of our scheme on constructing accurate classifiers. Given a testing data tuple t, the constructed classifier is used to predict a class label C_i that maximizes the probability of $t \in C_i$, which is

$$P(C_i \mid t)=\frac{P(C_i,t)}{P(t)}. \tag{3.36}$$

Since $P(t)$ is constant for all classes, the objective can be reduced to finding C_i that maximizes $P(C_i, t)$. Nonetheless, since t contains n attributes, the cost of computing $P(C_i, t)$ is still too expensive. Therefore, a commonly used compromise is to compute

$P(C_i, t)$ based on $P(C_i, t_s)$, where t_s is a small subset of the n attributes of t. For example, in the naïve Bayesian classification algorithm, $P(C_i, t)$ is approximated by the product of all $P(C_i, a_j)$, where $j \in [1, n]$. In the decision tree classification algorithm, $P(C_i, t)$ is approximated by $P(C_i, t_s)$ where t_s is a set of selected test attributes corresponding to the nodes in the decision tree.

For any data tuple t, let t_s be a sub-tuple that contains h attributes of t. We measure the accuracy of our scheme by the maximum estimate error of $P(C_0, t_s) - P(C_1, t_s)$ after perturbation. Let the value of $P(C_i, t_s)$ estimated from the perturbed dataset be $\tilde{P}(C_i, t_s)$. The estimate error of $P(C_0, t_s) - P(C_1, t_s)$ is defined as

$$l_e(h) = \max_{t, t_s} |(P(C_0, t_s) - P(C_1, t_s)) - (\tilde{P}(C_0, t_s) - \tilde{P}(C_1, t_s))|. \quad (3.37)$$

Based on these notions, we formally define the accuracy measure as follows.

Definition 3.3. The degree of error l_e for data classification is defined as the maximum value of $l_e(h)$ on all sizes h. That is,

$$l_e = \max_{h \in [1, n]} l_e(h). \quad (3.38)$$

Generally speaking, the degree of error measures the discrepancy between classifiers constructed from the original dataset and the perturbed dataset. Recall that μ is a predetermined parameter used by the data warehouse server to compute the current system perturbation level k^*. Let there be $A = T_0'T_0 - T_1'T_1$. We now derive an upper bound on the degree of error as follows.

Theorem 3.3. With our scheme, when m is sufficiently large, we have

$$l_e \le \frac{2\mu\sigma_1}{m}, \tag{3.39}$$

where σ_1 is the largest eigenvalue of A.

Proof: We first prove that in the cases where h is 1 or 2, there is $l_e(h) \le 2\mu\sigma_1/m$. The extension to the cases where $h \ge 3$ can be proved in analogy to the extension (from $|B| \le 2$ to $|B| > 3$) shown in the proof of Theorem 3.2.

Similar to the proof of Theorem 3.2, we first introduce an intuitive explanation of matrix A. Consider an element of A with indices $2i$ and $2j$, respectively. We have

$$\langle A \rangle_{2i,2j} = \sum_{t \in C_0} \langle t \rangle_{2i} \langle t \rangle_{2j} - \sum_{t \in C_1} \langle t \rangle_{2i} \langle t \rangle_{2j} = \#\{C_0, a_i = a_j = 1\} - \#\{C_1, a_i = a_j = 1\}, \tag{3.40}$$

where $\#\{C_q,\ a_i = a_j = 1\}$ is the number of data tuples in C_q that satisfy $a_i = a_j = 1$. Note that for all a_i and a_j, there is

$$\Pr\{C_q, a_i = b_1, a_j = b_2\} = \frac{\#\{C_q, a_i = b_1, a_j = b_2\}}{m}. \tag{3.41}$$

Thus, we have

$$\langle A \rangle_{2i-1+b_1, 2j-1+b_2} = m \cdot (\Pr\{C_0, a_i = b_1, a_j = b_2\} - \Pr\{C_1, a_i = b_1, a_j = b_2\}). \tag{3.42}$$

As we can see, $l_e(1)$ is in proportion to the maximum error on the estimate of the diagonal elements of A, and $l_e(2)$ is in proportion to the maximum error on the estimate of non-diagonal elements of A. Let the matrix of perturbed data be T_R. Let the

corresponding A derived from T_R be A_R. We now derive an upper bound on $\max_{ij}|\langle A - A_R\rangle_{ij}|$.

Recall that in the first step of the perturbation, a data provider computes $\tilde{t}_i = t_i V_k^* V_k^{*\prime}$. Let $\tilde{T}$ be an $m \times 2n$ matrix composed of $\tilde{t}_i$ (i.e., $\tilde{T} = [\tilde{t}_1; \ldots; \tilde{t}_m]$). $\tilde{T}_i$ and $\tilde{A}_i$ are defined correspondingly. In our scheme, V_k^* contains the first k^* eigenvectors of the current value of A^*. For the sake of simplicity, we consider V_k^* as the first k^* eigenvectors of A. In real cases, the first k^* eigenvectors of A^* converge to those of A fairly quickly.

Let Σ_k^* be a $k^* \times k^*$ diagonal matrix with diagonal elements be the first k^* eigenvalues of A (i.e., the diagonal vector of Σ_k^* is $[\sigma_1, \ldots, \sigma_{k^*}]$. We have

$$\tilde{A} = \tilde{T}_0'\tilde{T}_0 - \tilde{T}_1'\tilde{T}_1 = V_k^* V_k^{*\prime}(T_0'T_0 - T_1'T_1)V_k^* V_k^{*\prime} = V_k^* \Sigma_k^* V_k^{*\prime}. \quad (3.43)$$

That is, $\tilde{A}$ is the k^*–truncation of A [28]. As such, $\tilde{A}$ is the optimal rank-k^* approximation of A in the sense that within all rank-k^* matrices, $\tilde{A}$ has the minimum $\left\|A - \tilde{A}\right\|_2$. In particular, we have

$$\left\|A - \tilde{A}\right\|_2 = \sigma_{k^*+1}. \quad (3.44)$$

As we can see from the determination of system disclosure level k^*, our scheme maintains a cutoff k^* such that $\sigma_{k^*+1} \le \mu\sigma_1$. Thus, we have

$$\left\|A - \tilde{A}\right\|_2 \le \mu\sigma_1. \quad (3.45)$$

Since the absolute value of each element in an matrix is no larger than the 2-norm of the matrix [28], we have

$$\max\left|\left\langle A-\tilde{A}\right\rangle_{ij}\right| \le \left\|A-\tilde{A}\right\|_2 \le \mu\sigma_1. \tag{3.46}$$

As we can see from the computation of $R(t)$, for any $i, j \in [1, 2n]$, we have

$$\left\langle \tilde{t}\right\rangle_i^2 = \mathrm{Exp}\left(\left\langle R(t)\right\rangle_i^2\right), \tag{3.47}$$

and

$$\mathrm{Exp}\left(\left\langle t\right\rangle_i \left\langle t\right\rangle_j - \left\langle R(t)\right\rangle_i \left\langle R(t)\right\rangle_j\right) \le 2\left(\left\langle t\right\rangle_i \left\langle t\right\rangle_j - \left\langle \tilde{t}\right\rangle_i \left\langle \tilde{t}\right\rangle_j\right), \tag{3.48}$$

where Exp(·) refers to the expected value. When m is sufficiently large, we have $l_e(h) \le 2\mu\sigma_1/m$ for $h \le 2$. In analogy to the extension from $|B| \le 2$ to $|B| > 3$) in the proof of Theorem 3.2, we can prove that $l_e(h) \le 2\mu\sigma_1/m$ for all $h \in [1, n]$. □

As we can see from the theorem, the upper bound on the degree of error is in proportion to parameter μ. Thus, the data warehouse server can always control the accuracy of constructed classifier by adjusting μ.

III.7 Experimental Results

We now experimentally evaluate the performance of our scheme on real-world datasets. In particular, we compare the performance of our scheme with that of the randomization approach.

III.7.1 Association Rule Mining

We first compare the performance of our scheme with that of the randomization approach in association rule mining. We use a real dataset "BMS Webview 1" from Blue Martini Software. The dataset contains several months' click stream data from Gazelle.com, a leg-care web retailer that no longer exists. We choose this dataset because it has been extensively used (e.g., in KDD Cup 2000) to test the real-world performance of association rule mining algorithms [64]. The dataset includes 59,602 transactions and 497 items. The maximum transaction size (i.e., number of items in a transaction) is 267. There is no missing value in the dataset.

In order to demonstrate the effectiveness of our scheme on mining frequent itemsets with different sizes, we set different minimum support thresholds (*min_supp*) for different sizes of itemsets. In particular, we set *min_supp* = 3% for 1-itemsets (i.e., itemsets with one item), *min_supp* = 0.75% for 2-itemsets, and *min_supp* = 0.5% for itemsets with three or more items.

In order to make a fair comparison between our scheme and the randomization approach, we measure the accuracy of mined frequent itemsets by an extensively used real-world accuracy measure: the number of false positives and false negatives. False positives are infrequent itemsets that are misidentified as frequent. False negatives are frequent itemsets that are not identified (as frequent) correctly. We first compute the number of real frequent itemsets (represented by *FR*), the number of false positives (represented by *FP*), and the number of false negatives (represented by *FN*). Then, we compute (*FP* + *FN*) / *FR* to measure the error of mined frequent itemsets.

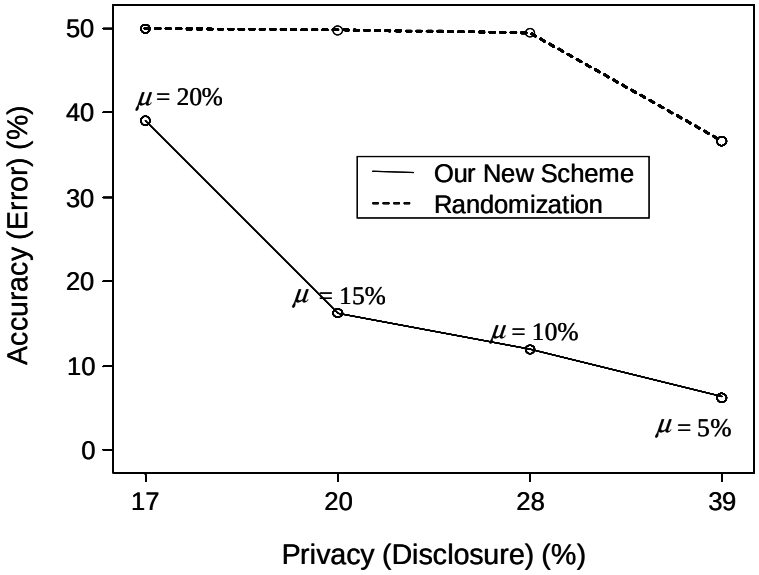

Fig. 4. Performance Comparison in Association Rule Mining.

To measure the level of privacy protection, we use the degree of privacy disclosure as defined in Definition 3.1. Recall that our scheme has a parameter μ, which is determined by how accurate the data warehouse server wants the constructed data mining model to be. We test the performance of our scheme when μ = 5%, 10%, 15%, and 20%, respectively. The initial value of T^* contains 497 transactions randomly generated such that an item has probability of 0.5% to be included in a transaction in T^*. The data warehouse server updates V_k^* when every 500 transactions are received. The data warehouse server directly uses the received (perturbed) transactions as input to the association rule mining algorithm, which is the a priori algorithm in our experiment [29].

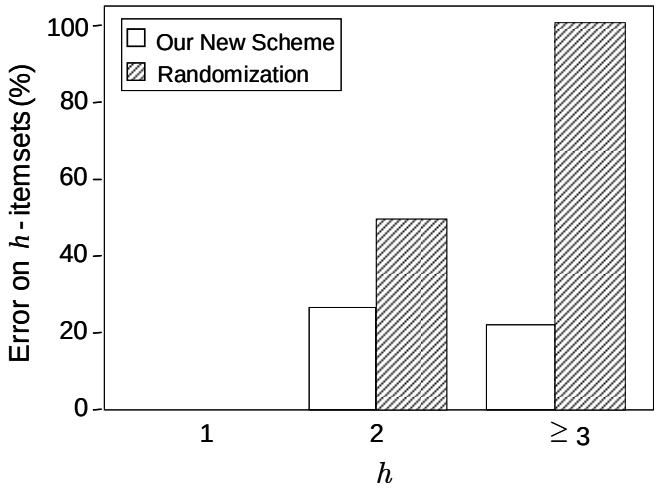

Fig. 5. Accuracy of Itemsets with Different Sizes.

We compare the performance of our scheme with that of the randomization approach in Figure 4 and Figure 5. As we can see from Figure 4, our scheme achieves a better tradeoff between privacy and accuracy at every configuration (from μ = 5% to μ = 20%). In order to demonstrate the effectiveness of our scheme on mining frequent itemsets with different sizes, in Figure 5, we compare the accuracy of our scheme with that of the randomization approach on mining frequent itemsets with 1, 2, and more than 3 items, respectively, while maintaining a privacy disclosure level of 20%. As we can see, while both approaches perform perfectly on 1-itemsets, our scheme outperforms the randomization approach on 2-itemsets and itemsets with 3 or more items.

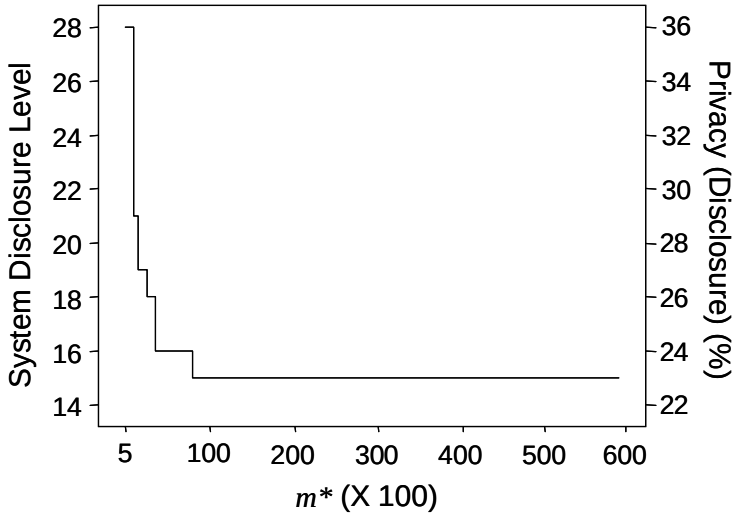

Fig. 6. System Disclosure Level in Association Rule Mining.

In Figure 6, we demonstrate the relationship between the system disclosure level and the number of transactions received by the data warehouse server. As we can see from Figure 6, the system disclosure level k^* decreases rapidly when more transactions are received. In the experiment, we set μ = 10%. Due to the results, the degree of privacy disclosure decreases to less than 30% when 2.5% transactions are received, and decreases further to less than 25% when 5.9% transactions are received.

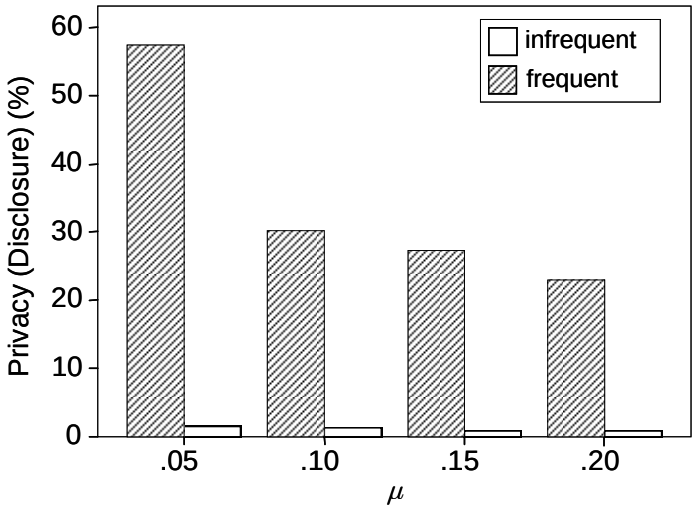

Fig. 7. Demonstration of Minimum Necessary Rule.

In Figure 7, we show that our scheme tends to disclose only those items that are most necessary for data mining. In particular, we compare the degree of privacy disclosure on items that have support less than 0.5% with items that have support larger than or equal to 0.5% when μ = 0.05, 0.10, 0.15, and 0.20, respectively. As we can see from the experimental setting, items with support less than 0.5% will not appear in any frequent itemset, thus can be considered as "less necessary" for association rule mining. Note that when the randomization approach is used, all items have the same (expected) degree of privacy disclosure. As we can see from Figure 7, when our scheme is used, the amount of information disclosed about less necessary (i.e., infrequent) items (i.e., items with support less than 0.5%) is much less than information disclosed about (frequent) items that are most necessary for association rule mining.

III.7.2 Data Classification

We first compare the performance of our scheme with that of the randomization approach in data classification. Then, we present the simulation results of our scheme on a real dataset. In order to make a fair comparison between the performance of our scheme and that of the randomization approach, we use the exact same training and testing datasets as in [4]. Please refer to [4] for a detailed description of the training dataset and the classification functions. The training dataset contains 100,000 data tuples. The testing dataset contains 5,000 data tuples. Each data tuple has nine attributes, including seven continuous attributes and two categorical ones. Five widely-varied classification functions are used to measure the tradeoff between accuracy and privacy in different circumstances. The randomization approach used is a combination of ByClass distribution reconstruction algorithm with Gaussian randomization operator [4], which performs the best in our experiment compared with other combinations proposed in [4] (i.e., combination of ByClass or Local algorithm with uniform or Gaussian distribution). We use the same classification algorithm, ID3 decision tree algorithm, as in [4].

Since our scheme assumes the dataset to contain only categorical data, we first need to transform continuous attributes to categorical ones. In order to do so, we split the possible values of each continuous attribute into four intervals based on its 1st quartile (i.e., 25% percentile), median, and 3rd quartile (i.e., 75% percentile). As such, each continuous attribute is transformed to a categorical attribute with 4 distinct values. Since the two categorical attributes have 5 and 9 distinct values, respectively, each private data

tuple is represented by a 42-dimensional binary vector ($\Sigma_j s_j = 4 \times 7 + 5 + 9 = 42$) after the discretization process.

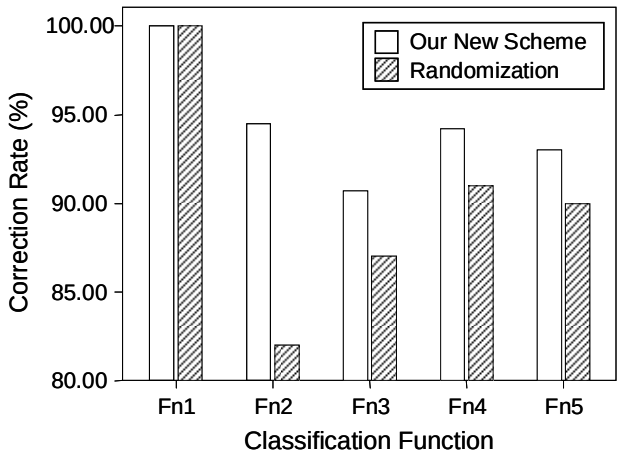

Fig. 8. Performance Comparison in Data Classification.

To evaluate the accuracy of constructed classifiers, we use a real-world accuracy measure: the correction rate of testing data tuples classified by the decision trees constructed by our scheme and the randomization approach. Figure 8 shows the comparison of correction rate while fixing the expected degree of privacy disclosure at 25%. In the figure, we use Fn*i* to represent the *i*-th classification function. Since our scheme allows different data providers to choose different levels of privacy disclosure, we compute the expected degree of privacy disclosure as the average value for all data providers. In our scheme, the data warehouse server updates the system disclosure level k^* and the perturbation guidance V_k^* once 100 data tuples are received. As we can see, while both approaches perform perfectly on Function 1, our scheme outperforms the randomization approach on the other four functions.

To demonstrate that our scheme is transparent to the classification algorithms, we also implement our scheme with the naïve Bayesian classification algorithm, and evaluate the performance on a real dataset. In particular, we use the congressional voting records database from the UCI machine learning repository [10]. The original data source is the Congressional Quarterly Almanac, 98th Congress, 2nd session, 1984. The dataset was donated by Jeff Schlimmer in 1987. The dataset includes 16 key votes for each member of the U.S. House of Representatives. It includes 435 records with 16 attributes (all of which are binary) and a class label describing whether the congressman is a democrat or republican. There are 61.38% democrats and 38.62% republicans in the dataset. The goal of classification is to determine the party affiliation based on each member's votes. There are 392 missing values in the dataset, which we substitute with values chosen uniformly at random from 0 and 1.

Since each data tuple has 16 binary private attributes, each data tuple is represented by a 32-dimensional binary vector. We first apply naïve Bayesian classification algorithm on the original dataset to build a naïve Bayesian classifier. We then apply our scheme with 9 different degrees of privacy disclosure and build 9 classifiers on the perturbed datasets. Then, we apply the same testing dataset to all 10 classifiers and compare their predictive accuracy. The predictive accuracy of the classifier built on the original dataset is 90.34%. In order to demonstrate the role of disclosure level k_i in our scheme, we simulate our scheme when all data providers choose the same disclosure level $k_i = k^*$.

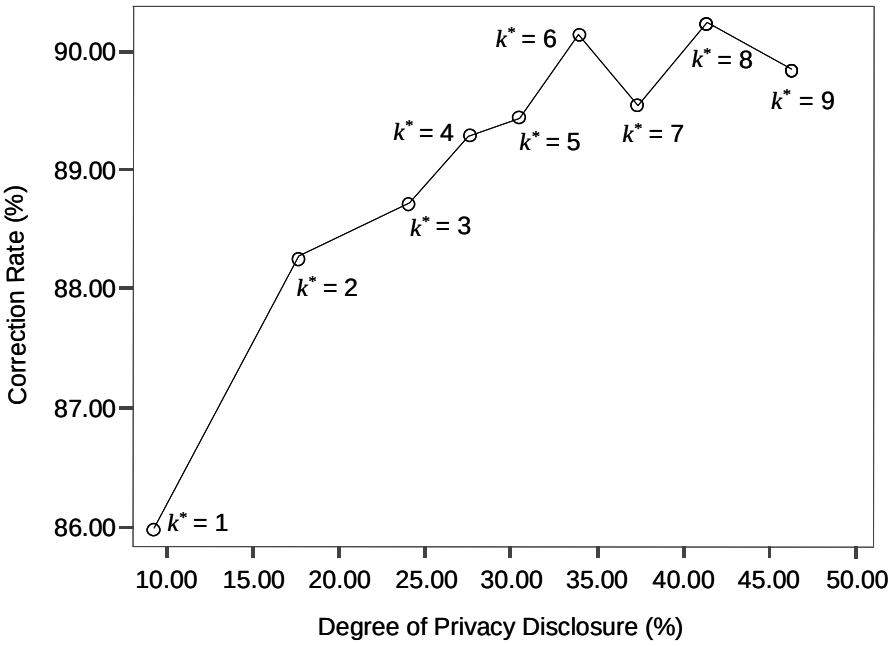

Fig. 9. Tradeoff between Accuracy and Privacy.

The predictive accuracy of classifiers built on perturbed datasets are shown in Figure 9. As we can see from the figure, when the degree of privacy disclosure is 9.56%, a naïve Bayesian classifier built on the perturbed dataset can predict the class label with correction rate of 85.99%. Thus, our classification can effectively preserve privacy while maintaining the accuracy of constructed classifiers.

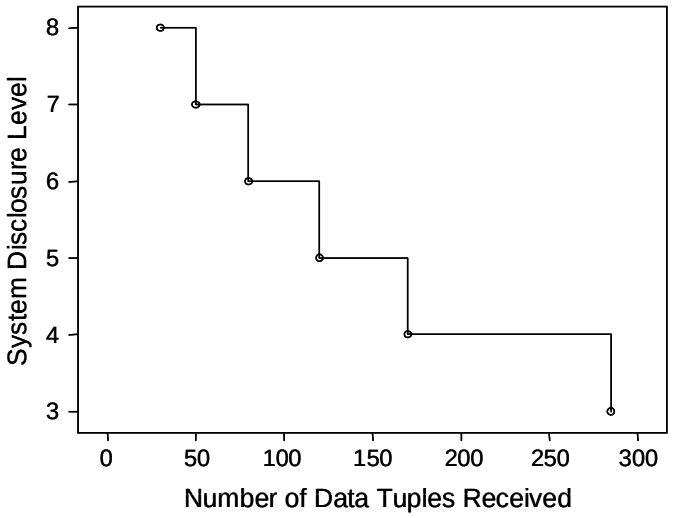

Fig. 10. System Disclosure Level in Data Classification.

To demonstrate that the system disclosure level k^* decreases rapidly during the collection of data tuples, we perform another simulation while fixing the parameter μ, which is used to compute k^*. Recall that generally speaking, the lower μ is, the more information is retained after perturbation. For a given μ, we investigate the change of k^* with the number of data tuples received by the data warehouse server (i.e., m^*). In most cases, k^* becomes very small fairly soon. For example, when μ = 15%, k^* decreases to be 2 after 50 data tuples are received. Figure 10 depicts the change of k^* with m^* when the degree of error is required to be very small (μ = 2.5%). As we can see, even when the error is strictly bounded, k^* still decreases fairly quickly when more data tuples are received.

III.8 Implementation

We present the implementation of our scheme and compare the runtime efficiency of our scheme with that of the randomization approach.

III.8.1 System Realization

A prototypical system for privacy-preserving data mining has been realized using our new scheme. The goal of the system is to deliver an online survey solution that preserves the privacy of survey respondents. The survey collector/analyzer and the survey respondents are modeled as the data warehouse server and the data providers, respectively. The system consists of a perturbation guidance component on web servers and a data perturbation component on web browsers. Both components are implemented as custom plug-ins that one can easily install to existing systems.

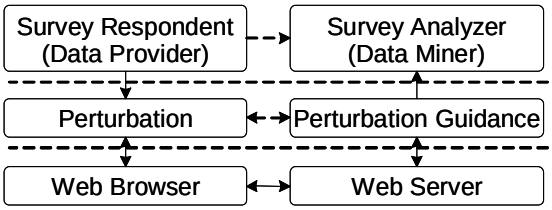

Fig. 11. System Implementation with Our Scheme.

As shown in Figure 11, the architecture of our system contains three separate layers: user interface layer, perturbation layer, and web layer. The top layer, or user interface layer, provides interface to the data providers (i.e., clients) and the data warehouse server. The middle layer, or perturbation layer, realizes our privacy-preserving scheme and exploits the bottom layer to transfer information. In particular, before the perturbation guidance V_k^* is received by a data provider, the data perturbation component first encrypts the private data and caches it on the client machine. When V_k^* is received, the data perturbation component decrypts the cached data, perturbs it based

on V_k^*, and transmits the perturbed data to the data warehouse server. The bottom layer, or web layer, consists of web servers and web browsers. As an important feature of our system, the details of data perturbation on the middle layer are transparent to both the data providers and the data warehouse server. Furthermore, the middle layer design on clients is independent of data mining tasks because the perturbation component of data providers does not vary for different data mining tasks.

III.8.2 Runtime Efficiency

We now compare the runtime efficiency of our scheme with that of the randomization approach. In particular, we compare the computational cost, space complexity, and communication overhead of both approaches, respectively. Without loss of generality, we make the comparison for privacy-preserving data classification problem.

For the randomization approach, we use the method proposed in [4] as an example. As other randomization-based methods, this method requires the original data distribution to be reconstructed before a decision tree classifier can be built on the randomized data. The distribution reconstruction process can be done by different algorithms. We use the ByClass reconstruction algorithm [4] as an example because it offers a tradeoff between efficiency and accuracy of reconstructed distribution. As stated in [4], the time complexity of the algorithm is $O(mn + nv^2)$ where m is the number of training data tuples, n is the number of private attributes in a data tuple, and v is the number of intervals on each attribute, which is assumed to be $10 \leq v \leq 100$ in [4]. Note that the computation-intensive part of randomization approach occurs on the critical time

path, as the distribution reconstruction has to be performed after all data tuples are collected and before a classifier can be constructed.

In our scheme, the perturbed data tuples are directly used to construct data mining models. The only computational overhead incurred on the data warehouse server is to update the system disclosure level k^* and the perturbation guidance V_k^*. The time complexity of each updating process is $O(n^2)$. As we mentioned in the design of our scheme, the data warehouse server may only need to update k^* and V_k^* once several data tuples are received. Since the number of attributes (i.e., n) is much less than the number of data tuples (i.e., m), the overhead of our scheme is significantly less than the overhead of the randomization approach. Besides, the computational overhead of our scheme does not occur on the critical time path. Instead, it occurs during the collection of data.

In terms of space requirement, the randomization approach only needs to keep in memory the number of perturbed data in each interval of each attribute. Thus, the space complexity of the randomization approach is $O(nv)$. Our scheme is space-efficient too. As we can see, the space needed by our scheme (to compute k^* and V_k^*) is $O(n^2)$. Since both n and v are smaller than m in most systems, the space requirement of either approach is insignificant.

In terms of communication efficiency, the randomization approach has the minimum possible communication overhead of $O(n)$ per data provider. For our scheme, the data providers and the data warehouse server needs to exchange the maximum acceptable disclosure level and the perturbation guidance. Nonetheless, since the disclosure level k^* is a small number (a heuristic average value of k^* is an order less than n) for most data

providers, the communication overhead ($O(nk)$ per data provider) of the exchange is insignificant. There may be concern on the upstream traffic (i.e., transmission of perturbed data) from the data providers to the data warehouse server when there are many distinct values for each attribute of the data tuple. In this case, the sparse nature of $R(t_i)$ provides an efficient way to encode $R(t_i)$ as a list of nonzero elements. Thereby, the overhead of transmitting $R(t_i)$ can be substantially reduced. For example, when the perturbed data tuple has the 9th and 12th bits equal to 1 and all other bits equal to 0, the data provider can transmit indices 9, 12 rather than the original binary tuple. If this encoding mechanism is used, the complexity of the upstream traffic is $O(n)$.

III.9 Summary

In this chapter, we propose a new scheme on privacy-preserving data mining and implement it to solve the privacy-preserving association rule mining and the privacy-preserving data classification problems. Compared with previous approaches, our scheme introduces a two-way communication mechanism between the data warehouse server and the data providers with little overhead. In particular, we let the data warehouse server send perturbation guidance to the data providers. Using this intelligence, the data providers perturb their data in a way that the private information transmitted to the data warehouse server is the most necessary part for data mining. As a result, our scheme offers a better tradeoff between accuracy and privacy. Also, we avoid the computationally expensive distribution reconstruction process on the critical time path, and distribute the computational load into the data collection process.

CHAPTER IV

INFERENCE CONTROL PROTOCOL*

In order to protect the private data stored in the data warehouse server, we need to control the private information disclosed to the data mining servers through answers to online analytical processing (OLAP) queries. Inference control protocol is designed for this purpose. Inference control protocol has two objectives: 1) to prevent a data mining server from inferring private information that it does not have the right to access, and 2) to answer queries (from data mining servers) that are necessary for data mining. In this chapter, we present the design of inference control protocols. As we mentioned in Chapter II, inference control protocol can be either query-oriented or data-oriented. Query-oriented protocols either accept a query or reject it, while data-oriented protocols generally add noise to query answers. In this study, we focus on query-oriented protocols due to the following reasons.

- As we mentioned in Chapter II, in the systems where original (unperturbed) data must be stored in the data warehouse server, data-oriented approach generally suffers from efficiency problems. Thus, query-oriented approach is a better choice for such systems.
- In the systems where data stored in the data warehouse server can be perturbed, our data collection protocol proposed in Chapter III can (by itself) guarantee the

---

* Part of the data reported in this chapter is reprinted with permission from "Cardinality-based Inference Control in OLAP Systems: An Information Theoretic Approach" by Nan Zhang, Jianer Chen, and Wei Zhao, 2004. *Proceedings of the ACM International Workshop on Data Warehousing and OLAP*, Pages 59-64. Copyright 2004 by ACM Press.

protection of individual private data. Recall that data collected by our data collection protocol can be directly used in data mining. Thus, the inference control protocol is no longer needed in such systems. We will exam the design of such systems more thoroughly in Chapter VI.

IV.1 System Model

We first introduce the system model for inference control protocol. Recall that in the system, there is one data warehouse server and multiple data mining servers. The data warehouse server is supposed to answer OLAP queries issued by the data mining servers on the data stored in the data warehouse. The purpose is for the data mining servers to construct accurate data mining models based on the query answers. The data warehouse server has no knowledge about the data mining tasks performed by the data mining servers.

Privacy concern exists in the system, as part of the data stored in the data warehouse can be private, and thus should not be accessed by the data mining servers. For each data mining server, there are certain data points in the data warehouse that it can access, and others that it cannot. Note that even for data points that a data mining server cannot access, the data mining server may still send OLAP queries over the aggregate of such inaccessible data in order to support data mining.

Due to the minimum necessary rule, the inference control protocol should only allow the data mining servers to learn the minimum private information necessary for data mining. Thus, the objective of the inference control protocol is to ensure that a data mining server cannot infer any inaccessible data point based on the query answers and

the data points it can access, while allowing the data mining servers to construct accurate data mining models.

IV.2 Performance Measurement

We now define the performance measures for inference control protocol. Due to the objectives of inference control protocol, the performance should be measured in terms of both privacy protection and support for data mining tasks (i.e., information availability to the data mining servers). We define the privacy disclosure level and the information availability level as the measures of privacy protection and information availability, respectively.

For each data point c in the data warehouse (i.e., each cell in the data cube), the privacy disclosure level of c is defined as the percentage of private information in c that is disclosed to a data mining server. Formally speaking, the definition of privacy disclosure level can be stated as follows.

Definition 4.1. *The privacy disclosure level μ_c on a data point c is defined as*

$$\mu_c = \frac{I(c;\Omega)}{H(c)}, \tag{4.1}$$

where $H(c)$ is the information entropy of c, Ω is the set of all query answers issued to a data mining server, and $I(c; \Omega)$ is the mutual information between c and Ω.

Generally speaking, the entropy $H(c)$ measures how much information is contained in c, and the mutual information $I(c; \Omega)$ measures how much information about c can be inferred from Ω. As such, the privacy disclosure level μ_c measures the percentage of private information in c that is disclosed to a data mining server.

Based on the privacy disclosure level, we can define the *maximum privacy disclosure level* μ, which is pre-determined by the data warehouse server as the maximum acceptable level of privacy disclosure. For all data points c in X, there is

$$\mu_c < \mu. \tag{4.2}$$

Recall that the data warehouse server needs to support various data mining tasks (and answer various queries) of the data mining servers. Therefore, we need to measure the information availability level based on specific queries requested by the data mining servers. Since we focus on query-oriented protocols, we measure the *information availability level* by the percentage of queries submitted by a data mining server that are correctly answered by the data warehouse server.

IV.3 Protocol Design

We first introduce some basic notions. Then, we present a cardinality-based inference control protocol that supports all common OLAP operations, and can protect private data from being compromised by the data mining servers.

TABLE 6

Example of Data Cube

Sales (× $100)	*Jan*	*Feb*	*Mar*	*Sum*
Book	192	Known	220	S_B
Used Book	Known	20	Known	S_U
CD	Known	30	87	S_D
Video	168	Known	96	S_T
Sum	S_1	S_2	S_3	

IV.3.1 Based Notions

Data stored in the data warehouse are organized in well-disciplined physical structures. A commonly used physical structure is data cube [29]. A 2-dimensional 4×3 data cube is shown in Table 6 as an example. Suppose that the data warehouse contains an *n*-dimensional $d_1 \times d_2 \times \cdots \times d_n$ data cube *X*. Each data point in the data cube is called a *cell* of *X*. Note that certain cells in the data cube are marked as "Known". These cells are the data points that can be accessed by the data mining servers. Since the data mining servers can freely access these data points, and may even learn such data from external knowledge, we follow a conventional (and conservative) security assumption that the data mining servers already know these data points as pre-knowledge.

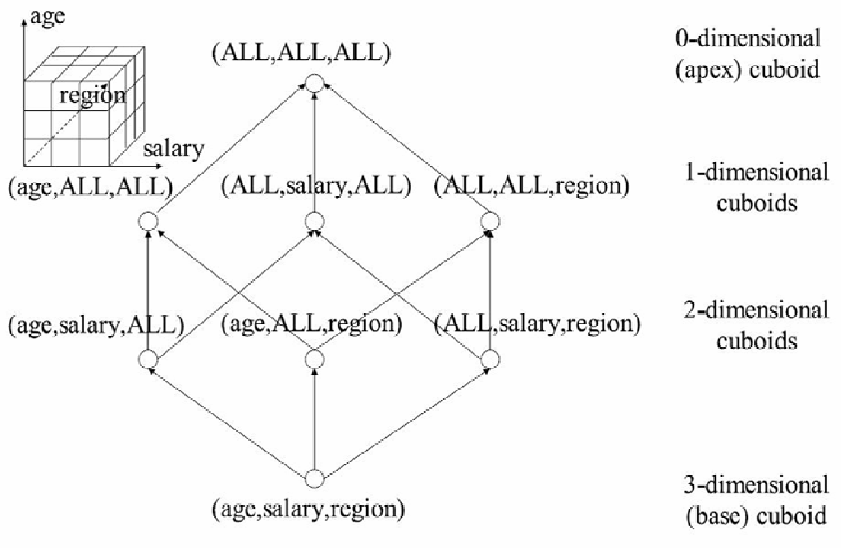

Fig. 12. Lattice of Cuboids.

Recall that in order to satisfy the online response time requirement, the OLAP queries on *X* are restricted to those that can be derived from the predefined *cuboids* of *X* [29].

Figure 12 shows a data cube and the lattice of cuboids of the data cube. Each cuboid shows a view of the data cube at different aggregation (i.e., group by) level. The lattice of these cuboids uniquely determines the schema of the data cube. In general, an *n*-dimensional data cube *X* has 2^n cuboids. A cuboid is *k*-dimensional if it is aggregated by *k* dimensions (i.e., group-by by *k* attributes). The 0-dimensional cuboid represents the aggregate of all data points in *X*, while each *n*-dimensional cuboid is a cell of *X*.

Based on the definition of cuboid, we can define an OLAP query Q_i by a 2-tuple $\langle P_i, C_i \rangle$, where P_i is the query operation (e.g., SUM) and C_i is the cuboid covered by the query. A query is *k*-dimensional if it covers an $(n - k)$-dimensional cuboid.

Given the cuboid definition, we now define the compromiseability of a cuboid. Generally speaking, the compromiseability of a cuboid *C* measures how difficult it is for a data mining server to compromise private information contained in the cuboid. The greater the compromiseability of *C* is, the more difficult it is for a data mining server to compromise a private cell in *C*.

Definition 4.2. Suppose that *C* is a *k*-dimensional cuboid in an *n*-dimensional data cube. The compromiseability of *C*, denoted by *l*, is defined as follows.

- If $k \geq n - 1$, the compromiseability of *C* is the number of private cells in *C*.
- If $k < n - 1$, the compromiseability of *C* is defined as

$$l = \min_{C_i \in \Omega} l_i. \tag{4.3}$$

  where l_i is the compromiseability of C_i, and Ω is the set of all $(k + 1)$-dimensional cuboids contained in *C* (i.e., connected to *C* in the lattice of cuboids).

IV.3.2 A Cardinality-based Protocol

We now present a cardinality-based inference control protocol. The protocol is cardinality-based in that it determines whether a query should be answered based on the number of private cells covered by the query. Before presenting the protocol, we first introduce the assumptions made in the design of the protocol. The correctness of the protocol is proved in the theoretical analysis part of the chapter.

In the design of this protocol, we make the following assumptions.

- The query operation is SUM, MIN, or MAX.
- The maximum privacy disclosure level $\mu = 1$. That is, a private data point is considered to be compromised by a data mining server if and only if the data mining server can infer the precise value of the data point.

As we can see from the first assumption, our protocol covers most of the commonly used operations in OLAP queries: SUM, MEAN (which can be derived from SUM), MAX, and MIN. The second assumption is a simplified assumption, as the value of μ is less than 1 in most real system. Nonetheless, we argue that this assumption is still reasonable in the design of inference control protocol due to the following reasons:

- Query-oriented inference control protocol is inherently over-conservative. Recall that as we mentioned in Chapter II, no unsafe query will be allowed by a query-oriented protocol. Nonetheless, queries that are indeed safe may often be rejected. Thus, even if we set the maximum privacy disclosure level to be 1, in most cases, the data mining server still cannot infer private information from the query answers.

- Realizing a system with $\mu < 1$ requires analysis on the values (rather than the schema) of data stored in the data warehouse server. Such analysis may result in an efficiency problem. Consider a 100×100 2-dimensional data cube. If the MAX queries on a row and a column return the same value, a data mining server may infer that with a fairly high probability, the cross cell of the row and the column is the answer of the queries. If $\mu < 1$, such inference is very likely to be considered as privacy breach. Nonetheless, in order to identify such privacy breach, one must analyze the values of the query answers as well as the values of the cells covered by the queries. Apparently, it will result in significant overhead to access the values of individual data at run-time. It remains a challenging problem to design an efficient inference control protocol that supports $\mu < 1$ and is capable of providing sufficient information availability.

Based on the assumptions, our inference control protocol is consisted of two steps: an offline pre-processing step and an online inference control step. The offline pre-processing step is performed before any query is received. In this step, the data warehouse server computes intermediate results that will be used in the online inference control step. After the completion of this step, when a query is received from a data mining server, the data warehouse server performs the online inference control step to determine whether to answer the query or to reject it. The algorithms of these two steps are shown in Table 7, respectively.

TABLE 7

A Cardinality-based Protocol

Offline pre-processing algorithm:
Input: an n-dimensional data cube X ($n \geq 2$), a predetermined compromiseability l_0.
Output: cardinality bounds b_j for all k-dimensional cuboids ($k \leq n-2$) in X.

1. **for** $k \leftarrow n$-2 down to 0 **do**
2. **for** every k-dimensional $d_1 \times \cdots \times d_k$ cuboid C_j in X **do**
3. compute l as the compromiseability of C_j;
4. $l = \max(l, l_0)$;
5. compute b_j as $b_j = l^{n-k-1} \sum_{i=1}^{n-k} d_i - (n-k)l^{n-k} - 1$;
6. **end for**
7. **end for**

Online query restriction algorithm:
Input: C_Q: cuboid covered by the recently received query Q, b_j computed offline.
Output: whether the query will be answered.
Note: the cardinality bound b of each cuboid is a global variable (i.e., change of b in each call will be kept in the future).

1. Let C_Q be (x_1, …, x_{n-k}, ALL, …, ALL).
2. Compute b_Q as the number of known cells aggregated in Q;
3. Compute l_Q as the compromiseability of C_Q;
4. **for** $i \leftarrow 1$ to $n-k-1$ **do**
5. Let C = (x_1, …, x_{i-1}, ALL, x_{i+1}, …, x_{n-k}, ALL, …, ALL);
6. Compute b as the (current) cardinality bound of C;
7. $b = b - b_Q$.
8. **if** $b \leq 0$ **or** $l_Q > l_0$ **then**
9. Reject the query;
10. **exit**;
11. **end if**
12. **end for**
13. Answer the query;

As we can see from the two algorithms, the data warehouse server maintains two data structures for each cuboid C in X: the compromiseability l and the cardinality bound b. Recall that as defined in Definition 4.2, the compromiseability measures how difficult it is for a data mining server to compromise private cells in C. The cardinality bound b is an upper bound on the number of known cells (i.e., non-private cells that can be accessed by the data mining server) in C, such that as long as the number of known cells

in *C* is less than or equal to *b*, no data point in *C* can be inferred from the set of *all* query answers on *X*. The correctness of *b* will be proved in the latter portion of the chapter.

The basic idea of our protocol can be stated as follows. First, we make an observation that if a known cell has never been included in any query, then a data mining server cannot take advantage of the known cell and infer any private data from it. Therefore, we call a cuboid safe if the number of known cells in the cuboid that have been covered by at least one query is less than the cardinality bound of the cuboid. Note that the safe state of a cuboid depends on the query history and may change when new queries are answered. Due to our definition of the cardinality bound, no privacy breach exist in a (currently) safe cuboid.

Another observation is that a *k*-dimensional cuboid is safe if every (*k* + 1)-dimensional cuboid it contains is safe. For example, in Figure 12, (age= 20, all, all) contains (age= 20, salary= 50, 000, all)). We will justify this observation in the theoretical analysis part of the chapter.

Based on these two observations, we introduce an offline pre-processing algorithm to the materialization of cuboids [29]. During the materialization of any *k*-dimensional cuboid *C*, we compute the cardinality bound *b* for *C*. When an *h*-dimensional query (on an (*n* – *h*)-dimensional cuboid C_Q) is received, our protocol does the following things.

1. Our protocol determines if the answer to *h* alone (i.e., without the help of query history) will allow a data mining server to infer private information from it[4].

[4] This is tacitly checked in Step 8 of the online query restriction algorithm by asserting $l_Q > l_0$.

2. If not, our protocol determines if every ($n - h - 1$)-dimensional cuboid that contains C_Q is safe. If all such cuboids are safe, then due to the above-mentioned first observation, all cuboids with dimensionality smaller than $n - h$ are safe. Thus, the query can be safely answered. Otherwise, the query is rejected

As we can see, the key issue in the design of our protocol is to properly decide the "cardinality bound" b so that more queries can be answered without privacy disclosure. Given an n-dimensional data cube X, we prove that a k-dimensional cuboid C is safe if it satisfies

$$b < l^{n-k-1}\sum_{i=1}^{n-k} d_i - (n-k)l^{n-k} - 1, \tag{4.4}$$

where $d_1, d_2, \ldots, d_{n-k}$ are the dimension domains of C (i.e., C is $d_1 \times d_2 \times \ldots \times d_{n-k}$), and l is the compromiseability of C. This result is used in Step 5 of the offline pre-processing algorithm. We will prove the correctness of this statement in the theoretical analysis. As we can see from (4.4), the result in [58] is actually a special case of our result when $l = 2$ and the dimensionality of the cuboid is equal to $n - 2$.

IV.4 Theoretical Analysis

We now prove the correctness of our inference control protocol. In order to do so, we first formulate the inference control problem in an information-theoretic manner. Readers unfamiliar with information theory are referred to the literature (e.g., [13]) for details.

Lemma 4.1. Given an n-dimensional data cube X, privacy breach occurs in X if and only if there is a private cell x_0 in X with

$$H(x_0 \mid \mathrm{AQ}) = 0, \tag{4.5}$$

where AQ is the set of answers to queries, and $H(x_0|\mathrm{AQ})$ is the conditional entropy of x_0 based on AQ.

We now transform the conditional entropy of x_0 to the conditional entropy of answers to queries, which is much easier to analyze. For a given cell x_0 in X, we have

$$H(x_0 \mid \mathrm{AQ}) = 0 \Leftrightarrow H(\mathrm{AQ}) - H(\mathrm{AQ} \mid x_0) = H(x_0). \tag{4.6}$$

Lemma 4.2. Privacy breach occurs in X if and only if there exists a private cell x_0 in X with

$$H(\mathrm{AQ}) - H(\mathrm{AQ} \mid x_0) = H(x_0). \tag{4.7}$$

For all possible data cubes X with t_0 known cells, Let $f_{\max}(t_0)$ and $f_{\min}(t_0)$ be the maximum and minimum entropies of answers to all queries on X, respectively. We have the following theorem

Theorem 4.3. Given an n-dimensional data cube X with t_0 known cells, no privacy breach occurs in X if for every private cell x_0, we have

$$f_{\max}(t_0) - f_{\min}(t_0 + 1) < H(x_0). \tag{4.8}$$

Proof: Due to Lemma 4.2, no privacy breach occurs if for every private cell x_0, there is

$$H(\mathrm{AQ}) - H(\mathrm{AQ} \mid x_0) < H(x_0). \tag{4.9}$$

Given x_0, we can construct a new data cube X' that is the same as X except that x_0 is known by the data mining servers. Let AQ′ be the set of answers to queries on X'. Due to the definition of conditional entropy, we have

$$H(\mathrm{AQ} \mid x_0) = H(\mathrm{AQ}'). \tag{4.10}$$

Due to the definition of $f_{\max}$ and $f_{\min}$, we have

$$H(\mathrm{AQ}) \leq f_{\max}(t_0). \tag{4.11}$$

$$H(\mathrm{AQ} \mid x_0) = H(\mathrm{AQ}') \geq f_{\min}(t_0 + 1). \tag{4.12}$$

In other words, no privacy breach occurs in X if for every private cell x_0, we have

$$f_{\max}(t_0) - f_{\min}(t_0 + 1) < H(x_0). \tag{4.13}$$

□

In order to simplify the discussion, hereafter we use $H(x)$ to denote the minimum entropy of all private cells. Due to the definition of information entropy, for any given query Q with operator SUM, MIN, or MAX, the entropy of answer to Q satisfies $H(x) \leq H(\mathrm{AQ}) \leq H(x) + O(\log |\mathrm{Q}|)$, where |Q| is the number of private cells covered by Q. Since $H(x)$ is the entropy of a private cell, we can safely assume that $H(x) \gg \log |\mathrm{Q}|$. In other words,

$$H(\mathrm{AQ}) \approx H(x). \tag{4.14}$$

We now derive an upper bound on $f_{\max}$ and a lower bound on $f_{\min}$ as follows.

Theorem 4.4. There is

$$f_{\max}(t) \leq \left(\sum_{j=1}^{n} d_j - n + 1 \right) H(x). \tag{4.15}$$

The theorem can be easily proved based on the properties of SUM, MIN, and MAX. In order to derive a lower bound on $f_{\min}$, we have the following theorem, which introduces a concept called maximum compromiseable data cubes.

Definition 4.3. (Maximum compromiseable Data Cube) An n-dimensional $d_1 \times d_2 \times \cdots \times d_n$ data cube X is maximum compromiseable if and only if there exists $0 < l \leq \lfloor (\min_i d_i)/2 \rfloor$, such that

1. all cells with *all* indices less than or equal to l are private.
2. all cells with *at least two* indices greater than l are private.
3. all other cells are known by the data mining servers

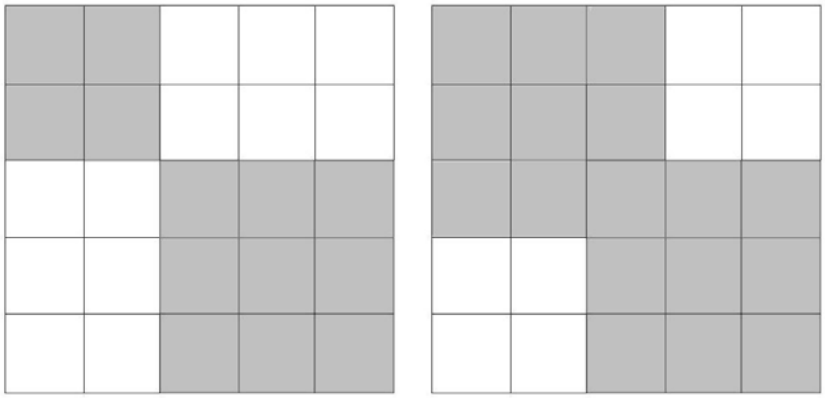

Fig. 13. Maximum Compromiseable Data Cubes.

As we can see, l is the compromiseability of X. Figure 13 shows two examples of maximum compromiseable data cubes. Based on the definition of maximum compromiseable data cube, we have the following theorem.

Theorem 4.5. Let X and X_0 be n-dimensional $d_1 \times d_2 \times \cdots \times d_n$ data cubes. Suppose that X_0 is a maximum compromiseable data cube with compromiseability of $l_0 > 1$. Let the number of known cells in X and X_0 be $t(X)$ and $t(X_0)$, respectively. If a data cube X satisfies

1. the compromiseability of X: $l = l_0$,

2. If $l < \min(d_1, \ldots, d_n)/2$, $t(X) < t(X_0) - 1$,
3. If $l \geq \min(d_1, \ldots, d_n)/2$, $t(X) \leq t(X_0)$,

then we have

$$H(\mathrm{AQ}) \geq \left(\sum_{j=1}^{n} d_j - 1 \right) H(x). \tag{4.16}$$

Proof: In order to simplify the discussion and help readers better understand our derivation, we prove this theorem on 2-dimensional data cubes. Readers may easily extend this proof to *n*-dimensional cases.

Suppose that a $d_1 \times d_2$ data cube X does not satisfy (4.16). In other words,

$$H(\mathrm{AQ}) < (d_1 + d_2 - 1) H(x). \tag{4.17}$$

Then, there exists a proper subset of AQ, denoted by AQ_0, such that

1. There exists a query answer in AQ_0, denoted by S, which can be derived from other query answers in AQ_0. In other words, there exists $S \in \mathrm{AQ}_0$ such that $H(S|\mathrm{AQ}_0 \backslash S) = 0$.
2. No such query answer exists in any proper set of AQ_0. In other words, for any $\mathrm{AQ}_0' \subseteq \mathrm{AQ}_0$, we have

$$\forall S \in \mathrm{AQ}_0', H(S \mid \mathrm{AQ}_0' \setminus S) > 0. \tag{4.18}$$

Without loss of generality, we assume that AQ_0 consists of $S_1, \ldots, S_{r1}$ and $S'_1, \ldots, S'_{r2}$ (this can be easily achieved by interchanging two rows or columns of X). A key point here is

Proposition 4.6. For all $i_1 > r_1$ and $j_1 < r_2$, $u_{i1,j1} = 1$. For all $i_2 < r_1$ and $j_2 > r_2$, $u_{i2,j2} = 1$.

Proof: Suppose there exist $i_1 > r_1$ and $j_1 < r_2$ such that $u_{i1,j1} = 0$. Then, we have mutual information

$$I(S'_{r2}; x_{i1,j1} \mid \mathrm{AQ}_0 \setminus S'_{r2}) > 0. \tag{4.19}$$

In other words, S'_{r2} cannot be derived from $\mathrm{AQ}_0 \backslash S'_{r2}$. For any other S that satisfies $H(S|\mathrm{AQ}_0\backslash S) = 0$, we have $H(S|\mathrm{AQ}_0\backslash S, S'_{r2}) = 0$. Nonetheless, this contradicts the condition that no such query answer exists in any proper set of AQ_0. Thus, there does not exist any $i_1 > r_1$ and $j_1 < r_2$ such that $u_{i1,j1} = 0$. Similarly, we can prove that for all $i_2 < r_1$ and $j_2 > r_2$, $u_{i2,j2} = 1$.

Based on the definition of compromiseability, we have $r_1, r_2 \geq l$. Due to Definition 4.3, if $l > \min(\lfloor d_1/2 \rfloor, \lfloor d_2/2 \rfloor)$, we have $t(X) > t(X_0)$. If $l \leq \min(\lfloor d_1/2 \rfloor, \lfloor d_2/2 \rfloor)$, we have

$$t(X) \geq r_1(d_2 - r_2) + r_2(d_1 - r_1) \geq k(d_1 + d_2) - 2k^2 = t(X_0). \tag{4.20}$$

However, this contradicts our assumptions that $t(X) < t(X_0) - 1$ when $l < \min(d_1, \ldots, d_n)/2$, and $t(X) \leq t(X_0)$ when $l \geq \min(d_1, \ldots, d_n)/2$. Thus, we have

$$H(\mathrm{AQ}) \geq (d_1 + d_2 - 1)H(x). \tag{4.21}$$

□

Theorem 4.6. Given an n-dimensional data cube X with compromiseability $l > 1$, no private data point in X will be compromised if $t(X) < l^{n-1} \Sigma d_j - nl^n - 1$, where $t(X)$ is the number of known cells in X.

Proof: We first consider the case where $l < \min(d_1, \ldots, d_n)/2$. Note that a maximum compromiseable data cube X_0 with compromiseability $l < \min(d_1, \ldots, d_n)/2$ satisfies $t(X_0) = l^{n-1} \Sigma d_j - nl^n$. Due to Theorem 4.5, for all data cube X with $t(X) < t(X_0) - 1$, we have

$$f_{\min}(t) \geq \left(\sum_{j=1}^{n} d_j - n + 1 \right) H(x). \tag{4.22}$$

Due to (4.15),

$$f_{\max}(t) \leq \left(\sum_{j=1}^{n} d_j - n + 1 \right) H(x). \tag{4.23}$$

Thus, for $t < l^{n-1} \Sigma d_j - nl^n - 1$, we have

$$f_{\max}(t_0) - f_{\min}(t_0 + 1) < H(x_0). \tag{4.24}$$

Due to Theorem 4.3, no privacy breach occurs in X. When $l \geq \min(d_1, \ldots, d_n)/2$, the theorem can be similarly proved using Theorem 4.3 and Theorem 4.5. □

IV.5 Summary

In this chapter, we propose an inference control protocol based on information-theoretic approach. In comparison with previous approaches, our scheme introduces an offline pre-processing algorithm which can be readily integrated into the materialization of cuboids. Our scheme also includes an online query restriction algorithm to enhance the support for data mining servers while maintaining data privacy. An upper bound on the cardinality of cells known by the data mining servers is derived to guarantee the safety of a data cube.

CHAPTER V

INFORMATION SHARING PROTOCOL[*]

In order to protect the privacy of data from being disclosed in information sharing, we need to control the private information shared between different data mining servers. Information sharing protocol is designed for this purpose. Information sharing protocol has two objectives: 1) to prevent a data mining server from compromising the private information of another data mining server, and 2) to allow the data mining servers to generate accurate information sharing results. In this chapter, we present the design and analyze the performance of information sharing protocols.

In this study, we focus on the two-party set intersection problem, in which two data mining servers collaborate to share the intersection of their datasets (i.e., local data mining models, e.g., association rules). Intersection is one of the most important problems in information sharing. Intersection protocols have been widely used as a primitive in many data mining applications including classification, association rule mining, etc. Nevertheless, we remark that our goal in this study is not to design solutions for specific information sharing problems. Rather, we are using the intersection problem as an example to demonstrate our methodology to deal with real-world adversaries in information sharing of privacy-preserving data mining systems.

---

[*] Part of the data reported in this chapter is reprinted with permission from "Distributed Privacy Preserving Information Sharing" by Nan Zhang and Wei Zhao, 2005. *Proceedings of the International Conference on Very Large Data Bases*, Pages 889-900. Copyright 2005 by ACM Press.

V.1 System Model

V.1.1 Parties

Let there be two data mining servers P_0 and P_1 in the system that we refer to as parties. In this chapter, unless otherwise indicated, we assume that P_1 intends to compromise the privacy of P_0 while P_0 does not have such intent. Thus, we call P_0 as the defending party and P_1 as the adversary. Neither party knows if the other party is an adversary.

Each party P_i has a private dataset V_i which contains numerous data points. Note that the word "data" in this chapter has a different meaning from that in previous chapters. In this chapter, we use data to represent the information that needs to be shared. In most cases, the data are the local data mining models, rather than the original data being mined (as in previous chapters).

Since the parties are supposed to share the intersection of their datasets, we assume that no data value appears more than once in the same dataset. As is commonly assumed in the literature, each data point in V_i is chosen independently and randomly from a (much larger) set V, which contains all possible values $v_1, \ldots, v_m$. We use p_{ij} to denote the probability that a data point v_j in V appears in V_i. For the simplicity of discussion, we assume that for all $0 \leq i \leq 1$ and $1 \leq j \leq m$, there is $p_{ij} = p$. Both parties know V and p. Nevertheless, neither party knows the size or content of the dataset of the other party.

V.1.2 Problem Statement

In an ideal situation, both parties should obtain $V_0 \cap V_1$ and nothing else at the end of the information sharing process. In reality, this requirement is often relaxed. A common compromise is to allow each party to learn the size of the dataset of the other party after

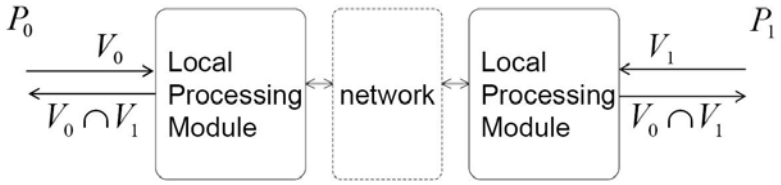

Fig. 14. System Infrastructure of Information Sharing.

information sharing [3]. As such, we say a system is *secure* if after information sharing, both parties obtain $V_0 \cap V_1$, the size of the dataset of the other party, and nothing else. We define the *privacy* of party P_0 as

$$V_0^{\mathrm{P}} = V_0 \setminus (V_0 \bigcap V_1) = V_0 \setminus V_1. \tag{5.1}$$

Note that when P_1 (maliciously) changes its input dataset to V_1', the privacy of P_0 does *not* change because we define V_0^{P} based on the *real* datasets instead of the *input* datasets. For example, when P_1 is a malicious adversary with no data to share (i.e., $V_1 = \varnothing$, where $\varnothing$ is the empty set), the privacy of P_0 should always be V_0 no matter what dataset P_1 manipulates to be its input to information sharing.

The objective of information sharing is to let both parties know $V_0 \cap V_1$ and protect V_0^{P} from being compromised by P_1.

V.1.3 System Infrastructure

There is an information sharing protocol jointly agreed by all parties. We assume that for each party, there is a local processing module that processes the dataset of the party and exchanges information with (the local processing module of) the other party. The information sharing protocol is implemented by the processing of and the communication between the local processing modules of the two parties. Figure 14

shows an information sharing system under this framework. As in common cases, we assume that the defending party will quit the protocol immediately if it can prove that the other party is an adversary.

Nevertheless, in order to model the threats from real-world adversaries, we do not impose any obligatory behavior restriction on either party. We say that a party changes its input dataset if the party manipulates a dataset as the input to its local processing module. We say a party revises its local processing module if the party deviates from the protocol by other means.

V.1.4 Strategies of Adversaries and Defending Parties

In the system, each party needs to choose

- the (possibly manipulated) input to its local processing module, and
- the (possibly revised) local processing module.

In addition, an adversary also needs to generate a reconstructed dataset $\tilde{V}_0$ that contains all data points the adversary believes to be in V_0^{P}.

As such, the attacking method of an adversary is to choose a combination of 1) the methods of manipulating its input dataset, 2) the modification of its local processing module, and 3) the generation of the reconstructed dataset. Since a defending party does not intend to compromise privacy, the defensive countermeasure of a defending party is limited to the former two methods. The attacking methods and defensive countermeasures will be further addressed in the latter portion of the chapter.

V.2 Performance Measurement

Given the attacking method and the defensive countermeasure, we need to measure the accuracy of information sharing result and the amount of privacy disclosure in information sharing.

V.2.1 Accuracy Measurement

We first present the accuracy measure of the system. Let s_A and s_D be the attacking strategy of the adversary and the defensive countermeasure of the defending party, respectively. We propose an *accuracy measure* $l_a(s_A, s_D)$ as follows to indicate the success of information sharing.

$$l_a(s_A, s_D) = \begin{cases} 1, & \text{if both party obtain } V_0 \cap V_1 \\ 0, & \text{otherwise} \end{cases} \tag{5.2}$$

Readers may wonder why we do not measure l_a continuously based on how many data points in $V_0 \cap V_1$ are successfully obtained by both parties, or how many parties successfully obtain $V_0 \cap V_1$. In other words, why do we consider partially accurate information sharing results as completely useless? It is true that in certain systems, partially accurate results are still valuable. We will discuss the extension of our results to these systems in the latter part of this chapter. However, we argue that for most cases, only the completely accurate information sharing results are desired by both parties. For example, consider the case where a government agency needs to intersect its list of (suspicious) terrorists with the passenger list of an airline company. One unidentified terrorist is enough for the failure of information sharing. In this case, partially accurate information sharing results are practically meaningless to the participating parties.

Furthermore, if partially accurate information sharing results are acceptable, the participating parties may use probabilistic approaches to randomize their data before sharing them, or share the statistical properties of their data, instead of using the information sharing approaches discussed in this chapter, which is generally more computationally expensive. Since in our problem statement, we assume that both parties want the *precise* $V_0 \cap V_1$, the discussion of such probabilistic approaches is out of the scope of our study. Readers may refer to [51] for related work in statistical databases.

V.2.2 Privacy Measurement

Recall that $\tilde{V}_0$ is the set of data points that the adversary believes to be in V_0^{P} and uses to perform unauthorized intrusion against the defending party. As such, a straightforward measure of privacy disclosure is the number of private data points in $\tilde{V}_0$. Let Exp[·] be the expected value of a random variable. Since $\tilde{V}_0$ may be randomly generated by the adversary, we formalize this measure as

$$\alpha(s_{\mathrm{A}}, s_{\mathrm{D}}) = \underset{\tilde{V}_0}{\mathrm{Exp}}\left[\frac{\left|\tilde{V}_0 \cap V_0^P\right|}{\left|V_0^P\right|}\right], \tag{5.3}$$

which is the expected percentage of private data points included in $\tilde{V}_0$. This measure is also referred to as *recall* in information retrieval [8]. Readers may raise a question of why we do not measure the maximum number of private data points in $\tilde{V}_0$, but measure the expected number instead. We believe that it is ineffective to measure the worst case situation. The reason is as follows: consider an attacking method which randomly generates $\tilde{V}_0$ from V. For any given system, it is always possible for the adversary to

generate $\tilde{V}_0 = V_0^P$. As such, the worst case privacy disclosure is always 100% of the private data.

Since the defending party may also change its input dataset, we note that there may also exist data points in $\tilde{V}_0$ which are not in V_0 (i.e., false positives). As such, there is another measure of privacy disclosure

$$\beta(s_A, s_D) = \underset{\tilde{V}_0}{\text{Exp}}\left[\frac{\left|\tilde{V}_0 \cap V_0^P\right|}{\left|\tilde{V}_0\right|}\right]. \tag{5.4}$$

This measure is also referred to as *precision* in information retrieval [8]. For the same reason as $\alpha(s_A, s_D)$, we measure the expected value instead of the worst case situation. Note that $\beta(s_A, s_D)$ is also important for measuring privacy disclosure because if only $\alpha(s_A, s_D)$ is used to measure the privacy disclosure, the maximum privacy disclosure (i.e., $\alpha(s_A, s_D) = 1$) can occur when the adversary generates $\tilde{V}_0 = V$.

As we can see, the amount of private information obtained by the adversary cannot be determined by either $\alpha(s_A, s_D)$ or $\beta(s_A, s_D)$ unitarily, but can be determined by the combination of them. This results in a problem comparing the amount of privacy disclosure in two cases if one case has a larger $\alpha(s_A, s_D)$ while the other one has a larger $\beta(s_A, s_D)$. Such comparison depends on the system setting, as is shown by the following example.

Suppose that the defending party always uses a countermeasure s_D. Let s_A be an attacking method with $\alpha(s_A, s_D) = 100\%$ and $\beta(s_A, s_D) = 30\%$. Let s'_A be an attacking method with $\alpha(s'_A, s_D) = 5\%$ and $\beta(s'_A, s_D) = 100\%$. We will show the comparison

between the amount of privacy disclosure when s_A and s'_A are used in two system settings.

First, consider a system where the two parties are two online retailers. The data points in V_i are the telephone numbers of the customers of P_i. The adversary uses the compromised telephone numbers to make unauthorized advertisement to the customers. In this system setting, the adversary prefers s_A because a wrong phone call (using v in $\tilde{V}_0$ but not in V_0^P) costs the adversary little. As such, s_A should have a higher privacy disclosure measure.

We now consider another system where the two parties are two consulting firms. Each data point in V_i is an unpublished profit expectation of a company. The adversary uses the compromised financial data to make investment on a high-risk stock market against the benefit of the defending party. The profit from a successful investment (using $v \in V_0^P$) is huge. Nonetheless, a failed investment (using v in $\tilde{V}_0$ but not in V_0^P) costs the adversary five times larger than the profit from a successful investment. In this system setting, the adversary prefers s'_A because if s_A is used, the expected return from an investment is less than 0 (i.e., the adversary would rather generate $\tilde{V}_0 = \varnothing$). Thus, s'_A should have a higher privacy disclosure measure in this system setting.

As we can see from the above example, we need to introduce the system setting to the measure of privacy disclosure. Let $\delta(v)$ be the profit obtained by the adversary from an unauthorized intrusion based on v ($v \in V$). Since the adversary intends to compromise the privacy of P_0, we have $\delta(v) > 0$ for all $v \in V_0^P$. Note that there must be $\delta(v) < 0$ for

all $v \notin V_0^{\mathrm{P}}$ because otherwise the adversary will always include such v ($v \notin V_0^{\mathrm{P}}$, $\delta(\mathrm{v}) \geq 0$) in $\tilde{V}_0$. We define system setting parameter μ as

$$\mu = \left| \frac{\mathrm{Exp}[\delta(v) \mid v \notin V_0]}{\mathrm{Exp}[\delta(v) \mid v \in V_0]} \right|. \tag{5.5}$$

Generally speaking, μ is the ratio between the loss of an adversary from a wrong guess of v ($v \notin V_0^{\mathrm{P}}$) and the gain of an adversary from a correct identification of v ($v \in V_0^{\mathrm{P}}$). Based on μ, we can derive a lower bound on $\beta(s_{\mathrm{A}}, s_{\mathrm{D}})$ to make $\tilde{V}_0$ meaningful for the adversary.

Theorem 5.1. *The profit obtained by the adversary from the unauthorized intrusion is no less than* 0 *if and only if*

$$\beta(s_{\mathrm{A}}, s_{\mathrm{D}}) \geq \frac{\mu}{\mu + 1}. \tag{5.6}$$

This theorem can be easily proved using our definition of μ. As we can see, when $\beta(s_{\mathrm{A}}, s_{\mathrm{D}}) < \mu / (\mu + 1)$, the return from unauthorized intrusion using $\tilde{V}_0$ is less than 0. Recall that $\varnothing$ is the empty set. Since the adversary is rational, the adversary prefers $\tilde{V}_0 = \varnothing$ (which leads to a profit of 0) to the $\tilde{V}_0$ generated by s_{A}. When $\tilde{V}_0 = \varnothing$, the amount of privacy disclosure is 0. As such, we define the *privacy disclosure measure* as follows.

$$l_{\mathrm{p}}(s_{\mathrm{A}}, s_{\mathrm{D}}) = \begin{cases} \alpha(s_{\mathrm{A}}, s_{\mathrm{D}}), & \text{if } \beta(s_{\mathrm{A}}, s_{\mathrm{D}}) \geq \mu/\mu + 1, \\ 0, & \text{otherwise.} \end{cases} \tag{5.7}$$

As we can see, the smaller $l_{\mathrm{p}}(s_{\mathrm{A}}, s_{\mathrm{D}})$ is, the less private data is obtained by the adversary and used to perform unauthorized intrusions against the defending party.

We now make a few remarks on the relationship between our privacy measure and the security models (e.g., statistically indistinguishable) commonly used in cryptography. The major difference is that while the commonly used security models measure whether the private information is *absolutely* secure against privacy intrusion, we intend to use a continuous value to measure the privacy protection level when absolute security cannot be achieved. As we will show in the latter portion of the chapter, when the adversary behavior is unrestricted, absolute security can only be achieved with expensive computational cost (for weakly malicious adversaries) or cannot be achieved at all (for strongly malicious adversaries). Thus, in order to design practical solutions to defend against such adversaries, we need to measure the amount of privacy disclosure by a continuous value.

V.3 Adversary Space

Recall that an adversary wants to compromise the private information of the other party and may or may not want to accomplish the information sharing (i.e., letting both parties know the intersection). Specifically, we assume that the objective of the adversary is to maximize the following objective function[5]

$$u_{\mathrm{A}}(s_{\mathrm{A}}, s_{\mathrm{D}}) = (1-\sigma) l_{\mathrm{a}}(s_{\mathrm{A}}, s_{\mathrm{D}}) + \sigma l_{\mathrm{p}}(s_{\mathrm{A}}, s_{\mathrm{D}}). \tag{5.8}$$

where $l_{\mathrm{a}}(\cdot)$ and $l_{\mathrm{p}}(\cdot)$ are defined in (5.2) and (5.7), respectively. Note that this model covers a wide range of adversaries. In the case where $\sigma = 1$, the adversary has no

[5] In (5.8), we do not consider adversaries with objective function $u_{\mathrm{A}}(s_{\mathrm{A}}, s_{\mathrm{D}}) = 0 \cdot l_{\mathrm{a}}(s_{\mathrm{A}}, s_{\mathrm{D}}) + 0 \cdot l_{\mathrm{p}}(s_{\mathrm{A}}, s_{\mathrm{D}})$ (i.e., adversaries that have neither interest on accurate information sharing results nor intent to compromise privacy). Since these adversaries are not threats of other parties' privacy, defending against them is out of the scope of this study.

interest in accomplishing the information sharing. When $\sigma = 0$, the adversary has no intent to compromise the other party's private information, and hence becomes a defending party. Generally speaking, the higher σ is, the more desire the adversary has to intrude privacy even at the expense of a failed information sharing. The lower σ is, the more desire the adversary has to share information rather than to compromise the privacy of the other party. In particular, we define two classes of adversaries based on the value of σ as follows.

Definition 5.1. An adversary is weakly malicious if and only if the adversary is not semi-honest and has $0 < \sigma < 1/2$. An adversary is strongly malicious if and only if the adversary is not semi-honest and has $1/2 \leq \sigma \leq 1$.

We now provide an intuitive explanation for our definition of weakly malicious adversaries. Consider the case when the information sharing fails. There is $l_a(s_A, s_D) = 0$. For a weakly malicious adversary, we have

$$u_A(s_A, s_D) = 0 + \sigma l_p(s_A, s_D) \leq \sigma < 1 - \sigma. \tag{5.9}$$

Note that $1 - \sigma$ is the value of $u_A(s_A, s_D)$ when both parties keep honest (i.e., the parties neither revise their local processing modules nor change their input datasets). Recall that we assume all parties make rational decisions to maximize their objective functions. Therefore, when the defending party is honest, a weakly malicious adversary will not intrude privacy if a successful intrusion of privacy will always result in at least one of the following two outcomes: 1) the adversary will be convicted as an adversary by the other party, or 2) at least one party cannot obtain $V_0 \cap V_1$.

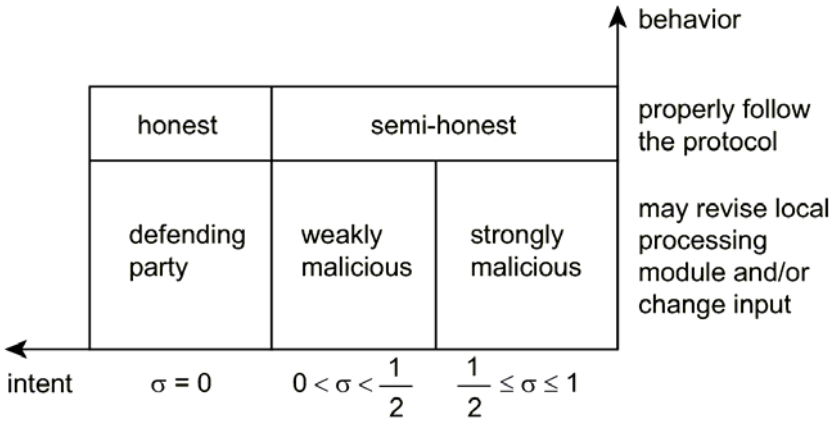

Fig. 15. Adversary Space.

With the introduction of weakly and strongly malicious adversaries, we can represent the population of adversaries in a two-dimensional space as is shown in Figure 15. Note that when $\sigma = 0$, the adversary is reduced to a defending party.

Given the adversary space, we consider the following two kinds of systems.

- Systems with weakly malicious adversaries. Adversaries in these systems are either semi-honest or weakly malicious.
- Systems with strongly malicious adversaries. Adversaries in these systems can be semi-honest, weakly malicious, or strongly malicious.

V.4 Design Goals

Before presenting our protocols, we first explain the design goals of information sharing protocols with different kinds of adversaries. We will analyze the design goals on the tradeoff between accuracy and privacy, as well as the design goals on efficiency (i.e., communication overhead), when the adversaries are semi-honest, weakly malicious, and strongly malicious, respectively.

V.4.1 Tradeoff between Accuracy and Privacy

We now show that absolute accuracy and security (which we will explain below) can be achieved when the adversaries are semi-honest or weakly malicious, but cannot be achieved if strongly malicious adversaries exist in the system.

When only semi-honest or weakly malicious adversaries exist in the system, there exist protocols which are strictly secure without loss of accuracy of information sharing result. Consider a protocol in which for each pair of data points in the two input datasets (i.e., $\langle v_0, v_1 \rangle$ where $v_0 \in V_0$ and $v_1 \in V_1$), the two parties call a protocol for Yao's millionaire problem [63] as a subroutine to determine if the data points are equal (i.e., if $v_0 = v_1$). If the data points are equal, the protocol adds the value of the data points into the intersection set. If the protocol for Yao's millionaire problem is secure against malicious adversaries, the intersection protocol is secure against weakly malicious adversaries (and thus semi-honest adversaries as well). Basically, the reason is that if only the adversary successfully compromises a private data point of the defending party, the information sharing result obtained by the defending party will always be wrong. As such, a weakly malicious adversary will choose a strategy to keep honest. As we can see, the protocol satisfies the following two conditions:

- (absolute accuracy) The optimal defensive countermeasure for the defending party is to keep honest. Thus, when both parties are defending parties, the information sharing always succeeds.
- (absolute security) After information sharing, the weakly malicious adversary P_1 obtains $V_0 \cap V_1$, $|V_0|$, and nothing else.

Given the presence of such protocol, the goal of a protocol designed for systems with semi-honest and/or weakly malicious adversaries is to protect privacy without loss of accuracy of information sharing result.

For strongly malicious adversaries, such a protocol does not exist. Consider a strongly malicious adversary with $\sigma = 1$. A possible (though not necessarily optimal) attacking method for the adversary is not to revise its local processing module, but always to insert one data point $v \notin V_1$ into its input dataset. Since the defending party does not know the exact size and content of V_1, either the malicious adversary compromises v when $v \in V_0$, or another honest party cannot obtain the correct information sharing result when it happens to have a dataset equal to $V_1 \bigcup v$. As such, when strongly malicious adversaries exist in the system, tradeoff has to be made between privacy protection and accuracy of information sharing result. Thus, the goal of designing protocols for systems with strongly malicious adversaries is *not* to achieve absolute accuracy and security, but to achieve an optimal tradeoff between privacy protection and accuracy of information sharing result.

V.4.2 Efficiency

We now derive lower bounds on the communication complexity of information sharing protocols that can achieve absolute accuracy and security with different kinds of adversaries. Formally speaking, a protocol achieves absolute accuracy and security if and only if it satisfies the following the following two conditions.

- If both parties follow the protocol properly without changing their input datasets, at the end of the execution of the protocol, both parties obtain $V_0 \cap V_1$, the dataset size of the other party, and nothing else,
- For any adversary, there is $l_p(s_A, s_D) = 0$, where s_A is the optimal attacking method for the adversary.

Theorem 5.2. If a protocol satisfies the above two conditions simultaneously, it has a communication overhead of at least $(|V_0'| + |V_1'|) \log(|V|)$ when semi-honest adversaries exist in the system, and at least $2(|V_0'| + |V_1'|) \log(|V|)$ when weakly malicious adversaries exist in the system. No such protocol exists when strongly-malicious adversaries exist in the system.

Proof: Note that the non-existence of protocols (absolutely) secure against strongly-malicious adversaries has been shown above. Thus, we will prove the theorem in two steps. First, we will show that a protocol for semi-honest adversaries has communication overhead of at least $(|V_0'|+|V_1'|) \log(|V|)$. Then, we will prove that a protocol for weakly malicious adversaries has communication overhead of at least $2(|V_0'|+|V_1'|)\log(|V|)$.

Let the message sent from the local processing module of P_i to that of the other party be t_i. Let r_i be the message P_i receives from the local processing module of the other party. The communication overhead of the protocol is $|t_0|+|t_1|$, where $|\cdot|$ is the length of a message. Since both parties obtain the intersection dataset, for each party P_i there is a function $h_i(r_i, V_i) = V_0' \cap V_1'$.

Let the Kolmogorov complexity [40] of a string x be $K(x)$. Roughly speaking, the Kolmogorov complexity of a string is the minimum number of bits in which the string can be compressed without information loss. Note that we can always consider a dataset or a message as a string. Since $|V| \gg |V_i|$, without loss of generality, we assume that $K(V_i') = |V_1'| \log(|V|)$. We first prove that when semi-honest adversaries exist in the system, in order to satisfy Condition 1, there must be $K(t_i) - K(t_i|V_i') \geq |V_i'| \log(|V|)$.

Suppose that $K(t_i) - K(t_i|V_i') < |V_i'| \log(|V|)$. For a given V_j', there must exist V_i', $V_i'' \subseteq V$, $V_i'' \neq V_i'$ such that t_i will be the same when P_i has a dataset of V_i' or V_i''. Without loss of generality, we assume that $V_i'' \not\subset V_i'$ (otherwise we can just exchange V_i' and V_i'' in the following proof). Consider the case when P_i has a dataset of V_i' (i.e., $V_i = V_i'$). Let $t_i = f(V_i', r_i)$. P_i can always find a dataset V_i'' which satisfies $f(V_i'', r_i) = f(V_i', r_i)$. That is, P_i can infer that for any $v \in V_i'' \setminus V_i'$, v is not in the dataset of the other party. This contradicts Condition 1.

Since $K(t_i) - K(t_i|V_i') \geq |V_i'| \log(|V|)$, the length of all messages transmitted is at least $(|V_0'| + |V_1'|) \log(|V|)$. That is, when semi-honest adversaries exist in the system, the communication overhead of a protocol that satisfies both conditions has communication overhead of at least $(|V_0'| + |V_1'|) \log(|V|)$.

We now prove that when weakly malicious adversaries exist in the system, in order to satisfy both conditions, there must be $K(t_i|V_i') \geq |V_j'| \log(|V|)$. For each party P_i, let the other party be P_j. We need to prove that $K(r_j|V_i') = K(t_i|V_i') \geq |V_j'| \log(|V|)$.

Suppose that $K(r_j|V_i') < |V_j'| \log(|V|)$. There must exist V_j', $V_j'' \subseteq V$, $V_j'' \neq V_j'$ such that for any V_i', r_j will not change when P_j substitutes its input from V_j' to V_j''. Again, without loss of generality, we assume that $V_j'' \not\subset V_j'$. Let $v \in V_j'' \setminus V_j'$. Consider the case when P_j (illegally) changes its input from V_j' to V_j'' and P_i strictly follows the protocol. If $v \notin V_i'$, then P_j can infer that $v \notin V_i'$ from $h_j(r_j, V_j'')$. If $v \in V_i'$, there are two possibilities. One is that $v \notin h_j(r_i, V_i)$. In this case, P_j successfully intrude privacy (by inferring that $v \in V_i'$) without changing the information sharing result. The other case is that $v \in h_j(r_i, V_i)$. This case is impossible as it requires P_i to infer that $v \in V_j''$ from a string with length $K(t_i|V_j')$ (since r_j is the same for V_j' and V_j''). If so, when V_j'' is the real dataset of P_j and $v \notin V_i'$, either P_j can infer that $v \notin V_i'$, or P_i can infer that $v \in V_j'$. Either case contradicts Condition 1. As we can see, when P_j changes its input from V_j' to V_j'' and P_i strictly follows the protocol, P_j can always intrude privacy without changing the information sharing result. This contradicts Condition 2.

As such, we have

$$K(t_i) = K(t_i) - K(t_i \mid V_i') + K(t_i \mid V_i') \geq (|V_0'| + |V_1'|)\log(|V|). \qquad (5.10)$$

To satisfy both conditions, the worst case communication overhead of the protocol must be greater than or equal to $2(|V_0'| + |V_1'|) \log(|V|)$. □

V.5 Protocol Design

We now propose protocols for systems with weakly malicious adversaries and strongly malicious adversaries, respectively. We remark that our goal in this study is not to promote specific protocols, but to demonstrate that when the adversary behavior is

unrestricted, simple solutions for information sharing problems still exist if we 1) constrain the adversary to be weakly malicious, or 2) make a tradeoff between accuracy and privacy.

V.5.1 Basic Tools

In both protocols, we use commutative encryption functions [20], [49] $E_0(\cdot)$ and $E_1(\cdot)$ on $v \in V$ that satisfy the following properties.

- E_i is computable in polynomial time. Given E_i, there exists a corresponding decryption function $D_i(\cdot) = E_i^{-1}(\cdot)$ which is also computable in polynomial time.
- E_0 and E_1 have the same value range. Suppose that c is chosen uniformly at random from the value range of $E_i(\cdot)$. For any v, $v' \in V$ which satisfies $v \neq v'$, no polynomial time algorithm A with time complexity $O(k)$ can generate output of 0 or 1 such that

$$\left| \Pr\{A(v, E_i(v), v', E_i(v')) = A(v, E_i(v), v', c)\} - \frac{1}{2} \right| > \frac{1}{poly(k)} \tag{5.11}$$

  where poly(·) is a polynomial function. Using the term in cryptography, we say that c and $E_i(v')$ is computationally indistinguishable given v, $E_i(v)$, and v'.
- $E_0(E_1(\cdot)) = E_1(E_0(\cdot))$.

An example of commutative encryption function is Pohlig-Hellman exponentiation cipher [45]

$$E_i(v) \equiv (h(v))^{e_i} \bmod p, \tag{5.12}$$

with the corresponding decryption function

$$D_i(c) = c^{d_i} \bmod p, \tag{5.13}$$

where p is a prime number, e_i and d_i are keys that satisfy $e_i \cdot d_i \equiv 1 \bmod (p\text{-}1)$, and h is a strong-collision-resistant hash function from V to all quadratic residues modulo p.

For a dataset V_i and encryption function E_i, we define $E_i(V_i)$ to be the set of $E_i(v \mid v \in V_i)$, which is represented by a sequence of all $E_i(v \mid v \in V_i)$ with lexicographical order. Given Property 3 of E_i, we have

$$E_1(E_0(V_0)) \cap E_0(E_1(V_1)) = E_0(E_1(V_0)) \cap E_0(E_1(V_1)) = E_0(E_1(V_0 \cap V_1)). \qquad (5.14)$$

V.5.2 Protocols

We now present our protocols designed for systems with weakly malicious adversaries and strongly malicious adversaries, respectively.

Since a party may change its input dataset, to avoid confusion, we use $\langle |V_i'|, V_i' \rangle$ (in contrast to the original dataset V_i) to denote the input from P_i to its local processing module. If a party P_i detects an inconsistency between the two inputs from the other party (thereby convicts the other party as an adversary[6]), the local processing module of P_i terminates execution immediately and quits the information sharing process.

---

[6] Note that a defending party will change neither V_i nor $|V_i|$ (as its inputs) because by changing them, the defending party may not obtain the accurate information sharing result.

TABLE 8

Protocol A: for Systems with Weakly Malicious Adversaries

1. Secretly exchange input dataset size $|V_0'|$ and $|V_1'|$,
2. If $|V_0'| > |V_1'|$, P_0 becomes P_s and P_1 becomes P_c, and vice versa. If $|V_0'| = |V_1'|$, P_0 and P_1 are assigned as P_s and P_c randomly.
3. P_s sends $E_c(V_c')$ to P_s,
4. P_s sends $E_s(E_c(V_c'))$ to P_c using the order of $E_c(V_c')$,
5. P_s sends $E_s(V_s')$ to P_c,
6. P_c computes $E_c(E_s(V_0' \cap V_1'))$. Since $E_s(E_c(V_c'))$ received by P_c in Step 4 is in the same order as $E_c(V_c')$ generated by P_c in Step 3, P_c can thereby finds the correspondingly $V_0' \cap V_1'$. P_c then sends $V_0' \cap V_1'$ to P_s.

TABLE 9

Protocol B: for Systems with Strongly Malicious Adversaries

1. Secretly exchange input dataset size $|V_0'|$ and $|V_1'|$,
2. Exchange encrypted input dataset $E_0(V_0')$ and $E_1(V_1')$,
3. Encrypt the received message and secretly exchange $E_0(E_1(V_1'))$ and $E_1(E_0(V_0'))$,
4. Each party now obtains $E_0(E_1(V_0' \cap V_1'))$ and decrypts it. Both parties exchange $E_1(V_0' \cap V_1')$ and $E_0(V_0' \cap V_1')$.

Table 8 and Table 9 show the pseudo-code for our Protocol A and Protocol B, which are designed for systems with weakly malicious adversaries and strongly malicious adversaries, respectively. In both protocols, we use a simultaneous secret exchange primitive which exchanges two secret messages from two (possibly malicious) parties such that either both parties know the secret of the other party, or no party can know the secret of the other party. This primitive has been realized by many protocols [11], [22], [42], [46].

V.6 Analysis of Protocols

We now analyze the performance of Protocol A and Protocol B respectively in terms of accuracy of information sharing results, protection of private data, and efficiency.

V.6.1 Analysis of Protocol A

We first show that Protocol A is secure when both parties are honest or semi-honest.

Theorem 5.3. When Protocol A is used, if both parties are honest or semi-honest, each party learns $V_0 \cap V_1$, the size of the dataset of the other party, and nothing else after information sharing.

Proof (Sketch): Since all parties follow the protocol strictly without changing their input datasets, we have $V_i' = V_i$.

In the protocol, P_s receives $|V_c|$, $E_c(V_c)$, and $V_0 \cap V_1$, P_c receives $|V_s|$, $E_s(E_c(V_c))$ and $E_s(V_s)$. We will prove that the view of either party in the protocol (the information it receives from the other party) is computationally indistinguishable from a view generated from its own dataset, $V_0 \cap V_1$ and the size of the dataset of the other party.

Let C be a sequence of $|V_c|$ lexicographically-ordered random variables chosen uniformly from the value range of $E_i(\cdot)$. We can construct a view $\langle |V_c|, C, V_0 \cap V_1 \rangle$ based on $|V_c|$ and $V_0 \cap V_1$. Due to Property 2 of $E_i(\cdot)$, $\langle |V_c|, C, V_0 \cap V_1 \rangle$ and $\langle |V_c|, E_c(V_c), V_0 \cap V_1 \rangle$ are computationally indistinguishable. Thus, P_s learns $V_0 \cap V_1$, $|V_c|$, and nothing else after information sharing.

We now construct a view to simulate the view of P_c. Let C_s be a set of $(|V_s| - |V_0 \cap V_1|)$ data points chosen uniformly from $V \backslash V_c$. Let E_s' be a commutative encryption function (whose key is) randomly generated such that E_c and E_s' also satisfy the three properties

as E_0 and E_1. We construct a view $\langle |V_s|, E_s'(E_c(V_c)), E_s'((V_0 \cap V_1) \cup C_s)\rangle$ based on $V_0 \cap V_1$, V_c, and $|V_s|$. Due to Property 2, the constructed view is computationally indistinguishable to $\langle |V_s|, E_s(E_c(V_c)), E_s(V_s)\rangle$. Thus, P_c learns $V_0 \cap V_1$, $|V_s|$, and nothing else after information sharing. □

We now analyze the cases where weakly malicious adversaries exist in the system. Let s_D^0 be a defensive countermeasure which will neither change the input dataset nor revise the local processing module (i.e., to keep honest). We derive an upper bound on the amount of privacy disclosure as follows.

Theorem 5.4. When the adversary is weakly malicious, let s_A be the optimal attacking method for the adversary. When Protocol A is used, there is $l_a(s_A, s_D^0) = 1$ (i.e., the information sharing always succeeds) and $l_p(s_A, s_D^0) \leq \sqrt{p/|V|}$, where p is the probability that a data point $v \in V$ appears in V_i.

Proof: Since the defending party keeps honest, we have $V_0' = V_0$. First, we show that the adversary cannot compromise any private information when it becomes P_s in the protocol. As we can see, P_s receives $E_c(V_c')$ in step 3 and $V_0' \cap V_1'$ in step 6. The adversary cannot compromise privacy from $E_c(V_c')$ due to the property of the encryption function $E_c(\cdot)$. We note that if P_1 can infer private information from $V_0' \cap V_1'$ (i.e., $V_0^P \cap (V_0' \cap V_1') \neq \varnothing$), the information sharing fails because P_0 does not obtain the correct intersection. Following the definition of weakly malicious adversary, P_1 would prefer keeping honest. Thus, the adversary cannot compromise any private information when it becomes P_s.

We now show that the adversary can only compromise private information in ((V_0'\$V_0$) $\cap$ V_1) when it becomes P_c. In the protocol, P_c sends out $E_c(V_c')$ in step 3 and $V_0' \cap V_1'$ in step 6. In order to compromise private information, P_c may perform either one or both of the following two intrusions: 1) changing its input dataset V_c, and 2) deviate from the protocol in step 6. After step 6, P_c does not receive any more information. Thus, the only private information P_c can obtain is ((V_c'\$V_c$) $\cap$ V_s). Note that if $V_c \subseteq V_c'$, P_c can still compute $V_c \cap V_s = V_c \cap (V_c' \cap V_s)$ and send this correct intersection set to P_s in step 6.

Since V_1' and $|V_1'|$ have to be consistent, the attacking method is to generate V_1' such that $V_1 \subseteq V_1'$. We now compute $l_p(s_A, s_D^0)$. Note that $|V_1'|$ has to be determined before $|V_0'|$ is known by P_1. Thus, the optimal $|V_1'|$ must maximize the expected value of $l_p(s_A, s_D^0)$ on all V_0. We have

$$l_p(s_A, s_D^0) = \Pr\{|V_1'| < |V_0|\} \mathrm{Exp}_{V_0}\left[\frac{|(V_1' \setminus V_1) \cap V_0|}{|V_0|}\right]. \quad (5.15)$$

With mathematical manipulations, we have $l_p(s_A, s_D^0) \le \sqrt{p/|V|}$. □

The above theorem indicates that when the defending party keeps honest, our protocol has little privacy leakage when weakly malicious adversaries exist in the system. In practice, $|V|$ can be in the order of 10^9 while $|V_i|$ is in the order of 10^3. In this case, the expected number of data points compromised by the adversary is in the order of $10^{-4.5}$ or less.

Theorem 5.5. The communication overhead of our protocol is ($|V_0| + |V_1| + \min(|V_0|, |V_0|) + |V_0 \cap V_1| + k$) $\log(|V|)$, where k is a constant value.

Compared to that of the most efficient existing protocol which is secure against semi-honest adversaries [3], the overhead of our protocol is only $k\log(|V|)$ more, which occurs in the first step.

We now compare the communication overhead of our protocol with that of the protocols which are both absolutely accurate and absolutely secure against weakly malicious adversaries. Recall that in Theorem 5.2, we derive a lower bound of $2(|V_0'| + |V_1'|)\log(|V|)$ on the communication complexity of such protocols. As we can see, when $|V_0|$ and $|V|$ are large, our protocol has a communication overhead substantially lower than these (absolutely accurate and secure) protocols (by $\max(|V_0|, |V_1|)\log(|V|)$) with little privacy disclosure introduced.

V.6.2 Analysis of Protocol B

We first show that Protocol B is secure when both parties are honest, semi-honest, or weakly malicious. Since no protocol can achieve both absolute accuracy and absolute security when strongly malicious adversaries exist, we analyze the tradeoff between accuracy and privacy when Protocol B is used in a system with strongly malicious adversaries.

Theorem 5.6. When Protocol B is used, if both parties are honest, semi-honest, or weakly malicious, each party learns $V_0 \cap V_1$, the size of the dataset of the other party and nothing else after information sharing.

Proof: We first consider the case when both parties are honest or semi-honest. In this case, since all parties follow the protocol strictly without changing their input datasets, we have $V_i' = V_i$. The protocol is symmetric in that each party learns exactly the same

information about the dataset of the other party. Without loss of generality, we consider the information obtained by P_1. P_1 receives $|V_0|$, $E_0(V_0)$, $E_0(E_1(V_1))$, and $E_1(V_0 \cap V_1)$ after information sharing. In the proof of Theorem 5.3, we proved that the view of $\langle |V_0|, E_0(V_0), E_0(E_1(V_1))\rangle$ can be simulated by a view constructed from $V_0 \cap V_1$, V_1, and $|V_0|$. As we can see, $E_1(V_0 \cap V_1)$ can also be generated from $V_0 \cap V_1$. Thus, the view of P_1 is computationally indistinguishable to a view constructed from $V_0 \cap V_1$, V_1, and $|V_0|$. As such, when the Protocol B is used, each party learns $V_0 \cap V_1$, the size of the dataset of the other party and nothing else after information sharing.

When weakly malicious adversaries exist in the system, an adversary P_i can only infer private information from the dataset it receives in step 4 (i.e., $E_i(V_0 \cap V_1)$). As we can see, both parties obtain $|E_0(E_1(V_0 \cap V_1))| = |V_0 \cap V_1|$ after step 3. As such, if the adversary can infer private information from $E_i(V_0 \cap V_1)$, it cannot obtain the correct intersection $V_0 \cap V_1$. Thus, when Protocol B is used, the system is secure against weakly malicious adversaries. □

Recall that when strongly malicious adversaries exist in the system, tradeoff has to be made between accuracy and privacy protection. In order to analyze such tradeoff, we propose a game theoretic formulation of the information sharing system as follows.

We model the information sharing system as a non-cooperative game [26] $G(S_A\ S_D, u_A, u_D)$ between the two parties where S_A and S_D are the set of (all possible) attacking methods and defensive countermeasures, respectively, and u_A and u_D are the utility functions (i.e., objective functions) for the adversary and the defending party, respectively. The game is non-cooperative as neither party knows whether the other

party is an adversary. The utility function for the adversary is the objective function we defined in Chapter V.II. In particular, for a strongly malicious adversary with $\sigma = 1$ we have

$$u_{\mathrm{A}}(s_{\mathrm{A}}, s_{\mathrm{D}}) = l_{\mathrm{p}}(s_{\mathrm{A}}, s_{\mathrm{D}}). \tag{5.16}$$

In order to define the utility function for the defending party, we first need to identify the goals of the defending party. The defending party has two goals in information sharing. One goal is to share information and obtain $V_0 \cap V_1$. We assume that the defending party has to guarantee a success probability of $1 - \varepsilon$ for the information sharing if the other party is also a defending party. The other goal is to prevent its private data in V_0 from being compromised by the adversary. As such, we define the utility function for the defending party as

$$u_{\mathrm{D}}(s_{\mathrm{A}}, s_{\mathrm{D}}) = \begin{cases} -\infty & \text{if } \Pr\{l_{\mathrm{a}}(s_{\mathrm{D}}, s_{\mathrm{D}}) = 0\} > \varepsilon, \\ -l_{\mathrm{p}}(s_{\mathrm{A}}, s_{\mathrm{D}}) & \text{otherwise,} \end{cases} \tag{5.17}$$

where $l_{\mathrm{a}}(s_{\mathrm{D}}, s_{\mathrm{D}})$ is the accuracy measure when both parties are defending parties.

Our goal is to derive the Nash equilibrium [26] of the game which contains both the optimal attacking method and the optimal defensive countermeasure. In order to do so, we need to formulate the space of all possible attacking methods and defensive countermeasures. Recall that both attacking methods and defensive countermeasures need to determine the (possibly changed) input dataset and the (possibly revised) local processing module. Besides, an attacking method also needs to generate $\tilde{V}_0$ based on the information obtained in information sharing. In the rest of our analysis, we first consider a simple case where neither attacking methods nor defensive countermeasures revise the

local processing module. We first derive the Nash equilibrium of the game based on this simple case. Then, we will prove that neither party can benefit from revising its local processing module. As such, the Nash equilibrium derived for the simple case will not change when the parties are allowed to revise their local processing modules.

Due to our classification of adversaries, a strongly malicious adversary has $1/2 \leq \sigma \leq 1$. Nevertheless, we consider the worst cases where the adversary has $\sigma = 1$. That is, the only goal of the adversary is to intrude the privacy of the defending party.

Since Protocol B is secure when the adversary is semi-honest, in order to compromise the privacy of the other party, the adversary must change its input dataset. Since the intersection set may contain data points manipulated by the defending party, the adversary also needs to decide if a data point in $V_0' \cap V_1'$ should be included in $\tilde{V}_0$. We analyze the attacking methods for determining V_1' and $\tilde{V}_0$ respectively as follows.

- Change input dataset. The adversary P_1 can compromise private information in V_0^P by changing its input dataset to V_1'. As we can see, if the defending party keeps honest, the adversary will obtain the private information in $V_0 \cap V_1'$ after information sharing. Due to Protocol B, $|V_1'|$ has to be determined before any information about V_0 can be obtained. Without loss of generality, we assume that $|V_1'|$ is a function of $|V_1|$, denoted by $k_1(|V_1|)$. Due to our system assumption, the adversary has no previous knowledge about any data point in V_0. As such, the optimal method for the adversary to generate V_1' is to choose V_1' randomly from $V \backslash V_1$. Without loss of generality, we model the attacking method on changing the input dataset as to determine $k_1(|V_1|)$.

- Generate $\tilde{V}_0$ from $V_0' \cap V_1'$. Since neither party may revise its local processing module, the only information that an adversary can obtain from information sharing is $V_0' \cap V_1'$. To benefit its own interest, the adversary has only two methods to generate $\tilde{V}_0$.
    - $\tilde{V}_0 = V_0' \cap V_1'$.
    - $\tilde{V}_0 = \varnothing$.

    That is, $\tilde{V}_0$ either contains all data points in the intersection set, or make the intersection set an empty set. This can be easily observed from the definition of $l_p(s_A, s_D)$.

The defensive countermeasure contains the method of changing the two inputs $\langle |V_0'|, V_0' \rangle$ to the local processing module. Due to the protocol, $|V_0'|$ has to be determined before any information about V_1 can be obtained. Without loss of generality, we assume that $|V_0'|$ is a function of $|V_0|$, denoted by $k_0(|V_0|)$. The only information that P_0 can obtain before choosing V_0' is the size of the input dataset of P_1. As such, we assume that V_0' is a function of V_0 and V_1' and is represented by $f(V_0, |V_1'|)$ where $f(V_0, |V_1'|) \subseteq V$ and $|f(V_0, |V_1'|)| = k_0(|V_0|)$. We model the defensive countermeasure as $\langle k_0(|V_0|), f(V_0, |V_1'|) \rangle$.

Let $h = \lfloor |V\_0|/\mu \rfloor + 1$. Let V_d be a dataset with the same distribution as V_i. Recall that p is the probability that a data point in V appears in V_i. We derive the Nash equilibrium of the game as follows.

Theorem 5.7. The optimal defensive countermeasure $\langle k_0(|V_0|), f(V_0, |V_1'|) \rangle$ is

$$k_0(|V_0|) = \begin{cases} |V_0| + h, & \text{if } h + g(|V_0| + h) \le \varepsilon |V| \\ |V_0|, & \text{otherwise} \end{cases} \tag{5.18}$$

$$f(V0, |V_1'|) = \begin{cases} V_0 \bigcup U(V \setminus V_0, h), & \text{if } k = |V0| + h, \\ V_0 & \text{if } k = |V_0| \text{ and } |V_1'| < N_S, \\ U(V_0, N_S |V_0| / |V_1'|) & \text{otherwise} \end{cases} \tag{5.19}$$

where $g(\cdot)$ satisfies

$$g(i) = \sum_{j=1}^{|V|} \Pr\{|V_d| = j\} \cdot \mathrm{Exp}\left[|f(V_d, i) \setminus V_d| + |V_d \setminus f(V_d, i)| \right], \tag{5.20}$$

$U(V, j)$ is the set of j data points chosen uniformly at random from V, and N_S is the largest integer that satisfies

$$g(|V_0|) + \sum_{j=N_S}^{|V|} \left[\frac{(p|V|)^j e^{-p|V|}}{j!} \cdot \left(1 - \frac{N_S}{k_0(j)} \right) \cdot |V_0| \right] \le \varepsilon |V|. \tag{5.21}$$

An optimal attacking method $k_1(|V_1|)$ is $k_1(|V_1|) = N_S$. The above optimal attacking method and defensive countermeasure form the Nash equilibrium of the game.

Proof: We will prove the theorem in three steps. First, we will prove that the error rate does not exceed the upper bound ε. Second, we will show that when $k_0(|V_0|) = |V_0| + h$, we have $l_p = 0$. In the last step, we will prove the optimality of the strategy when $k_0(|V_0|) = |V_0|$.

- Error rate is controlled below ε. We first consider the case when $k_0(|V_0|) = |V_0| + h$. In this case, no matter what $|V_1'|$ is, we have

  $$f(V_0, \cdot) = V_0 \bigcup U(V \setminus V_0, h). \tag{5.22}$$

  If P_1 is a defending party, since $|V| \gg |V_0|$, the error rate is

$$\varepsilon_0 \approx \frac{h + Exp\left[|V_1' \setminus V_1| + |V_1 \setminus V_1'|\right]}{|V|} \leq \varepsilon. \tag{5.23}$$

That is, the error rate is no more than ε. When $k_0(|V_0|) = |V_0|$, the error rate is

$$\varepsilon_0 \approx \frac{1}{|V|}\left(g(|V_0|) + \sum_{j=N_S}^{|V|} \left[\Pr\{|V_1| = j\} \cdot (1 - N_S / k_0(j)) \cdot |V_0|\right] \right) \leq \varepsilon. \tag{5.24}$$

Thus, the error rate is also no more than ε.

- When $k_0(|V_0|) = |V_0| + h$, $l_p = 0$. When $k_0(|V_0|) = |V_0| + h$, we have $V_0' = V_0 \cup U(V \setminus V_0, h)$. Recall that $\tilde{V}_0$ is either $V_0' \cap V_1'$ or an empty set. If $\tilde{V}_0 = V_0' \cap V_1'$, we have

$$\alpha(s_A, s_D) = \frac{|V_0 \cap V_0'|}{|V_0|} = 1, \tag{5.25}$$

$$\beta(s_A, s_D) = 1 - \frac{|V_0' \setminus V_0|}{|V_0'|} < \frac{\mu}{\mu + 1}. \tag{5.26}$$

As such, we have $u_A < 0$. Thus, the adversary has to choose $\tilde{V}_0 = \varnothing$. That is, we have $l_p = 0$.

- When $k_0(|V_0|) = |V_0|$, the attacking method and the defensive countermeasure form the Nash equilibrium.

The basic idea of the proof is to show that when the defending party does not change its defensive countermeasure, the adversary cannot compromise any more private information by using a manipulated dataset with size larger than N_S, which can be easily observed from $f(V_0, |V_1'|)$. When the adversary does not change its attacking method, the defending party cannot preserve more private information because otherwise the error

rate would be larger than ε. As such, the state defined in the theorem is a state where no party can benefit by changing its attacking method or defensive countermeasure unitarily. The detailed proof of this step is mainly mathematical manipulations. Thus, we omit the detailed proof here. □

V.6.3 Generalization to Complicated Methods

We now prove that when Protocol B is used, neither the adversary nor the defending party can benefit by revising its local processing module.

Theorem 5.8. When Protocol B is used, the adversary cannot increase the expected value of its utility function by revising its local processing module.

Proof: First, the adversary will not deviate from the protocol in step 1 and 2 because by doing so, the adversary is actually changing its input dataset. Recall that we assume all parties are rational. As such, the adversary will not revise step 4 either. The reason is that after this step, the adversary cannot obtain any more information about the dataset of the other party. We now show that the adversary will not deviate from the protocol in step 3.

In step 3, the adversary P_1 sends $E_1(E_0(V_0'))$ to the defending party P_0. P_0 then uses $E_1(E_0(V_0'))$ to compute

$$E_1(E_0(V_0')) \cap E_0(E_1(V_1')) = E_0(E_1(V_0' \cap V_1')), \tag{5.27}$$

which will be decrypted to $E_1(V_0' \cap V_1')$ and sent to P_1 in step 4. Thus, we only need to prove that by changing $E_1(E_0(V_0'))$, the adversary cannot increase

$$| E_1(V_0' \cap V_1') | = | E_0(E_1(V_0' \cap V_1')) | = | E_1(E_0(V_0')) \cap E_0(E_1(V_1')) |. \tag{5.28}$$

Recall that the adversary cannot change $|E_1(E_0(V_0'))|$ because by doing so, the defending party will detect an inconsistency between $|E_1(E_0(V_0'))|$ and $|V_0'|$ and quit the information sharing. As such, we need to prove that the adversary cannot change $E_1(E_0(v_0 \mid v_0 \in V_0'))$ to collide with $E_0(E_1(v_1 \mid v_1 \in V_0'))$. This can be inferred from property 2 of the commutative encryption function. □

Theorem 5.9. When Protocol B is used, the defending party cannot increase the expected value of its utility function by revising its local processing module.

Proof: First, the defending party will not deviate from the protocol in step 1 and 2 because it can change its input instead. We remark that the defending party also will not revise step 3 because by doing so, it cannot obtain the information sharing result (i.e., $V_0' \cap V_1'$). As such, we now prove that the defending party will not deviate from the protocol in step 4.

In step 4, the defending party P_0 sends $E_1(V_0' \cap V_1')$ to the adversary P_1. P_1 then decrypts $E_1(V_0' \cap V_1')$ to $V_0' \cap V_1'$, which is the result of information sharing. Since the defending party obtains $|V_1'|$ before step 2, we only need to prove that before step 4, the defending party does not know anything more than $|V_1'|$ about V_1'. If so, the defending party will not revise step 4. Rather, it will change its input in step 2.

As we can see, the defending party has received $E_1(V_1')$ and $E_1(E_0(V_0'))$ since step 2. Thus, we need to prove that given V_0', $E_0(\cdot)$, $|V_1'|$, $E_1(V_1')$, and $E_1(E_0(V_0'))$, there does not exist any polynomial time algorithm with time complexity $O(k)$ and output $v \in V$ such that

$$\left| \Pr\{v \in V_1'\} - \frac{|V_1'|}{|V|} \right| > \frac{1}{poly(k)}, \tag{5.29}$$

where $poly(\cdot)$ is a polynomial function. This can be inferred from property 2 of the commutative encryption function. □

V.7 Numerical Results

Numerical measurement has not been commonly used to demonstrate system security because all possible attacking methods cannot be exhausted in a simulation. Nevertheless, we propose to use numerical measurements in our case. The reason is that in the theoretical analysis, we already derive the Nash equilibrium of the game, which is a state where neither party can benefit by unitarily changing its attacking method or defensive countermeasure. The numerical results shown actually demonstrate the privacy disclosure in this state, and thus can be used to demonstrate the real privacy protection performance of systems using our protocols.

We evaluate the system performance in terms of the maximum expected number of private data compromised by the adversary, which is $l_p(s_A, s_D)$, where s_A and s_D are the optimal attacking strategy and the optimal defensive countermeasure, respectively. The error rate of information sharing when both parties are defending parties is fixed to be $\varepsilon = 0$ for systems with weakly malicious adversaries and $\varepsilon = 0.1$ for systems with strongly malicious adversaries. With $|V_0| = 100$, we demonstrate the relationship between the amount of privacy disclosure and the size of the population set (i.e., $|V|$).

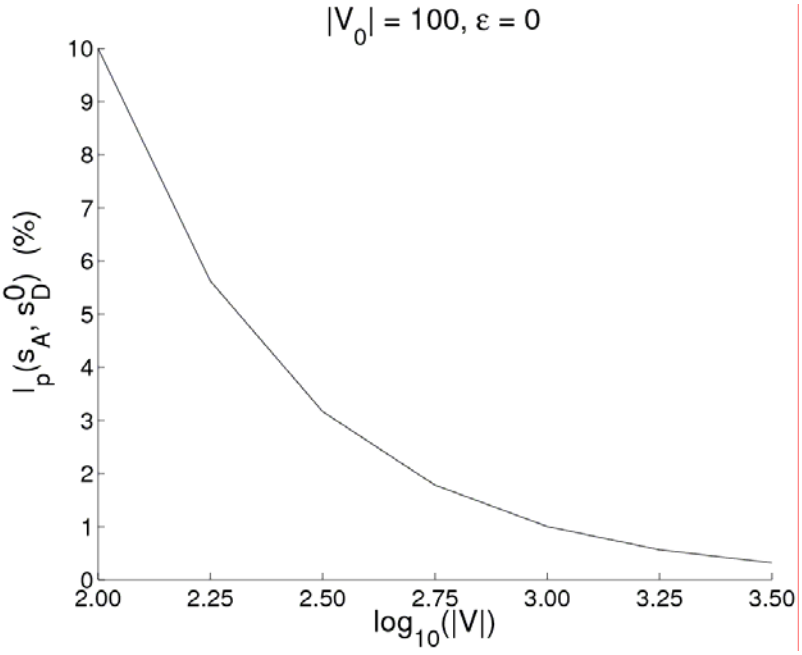

Fig. 16. Defense Against Weakly Malicious Adversaries.

For systems with weakly malicious adversaries, the maximum amount of privacy disclosure when Protocol A is used is shown in Figure 16. As we can see from the figure, the privacy leakage of our protocol is very small when $|V|$ is large. In particular, when $|V|$ is in the order of 10^3, the expected number of data points compromised by the adversary is no larger than 1.

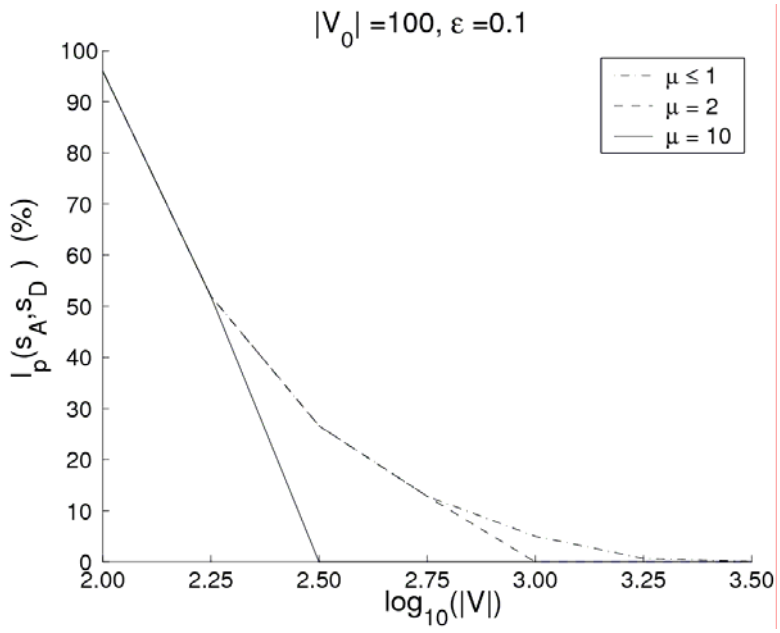

Fig. 17. Defense Against Strongly Malicious Adversaries.

For systems with strongly malicious adversaries, when Protocol B is used, the maximum amount of privacy disclosure is shown in Figure 17. As we can see from the figure, the higher $|V|$ or μ is, the less private data points are compromised by the adversary. In particular, no privacy disclosure occurs when $\mu \geq 2$ and $|V| \geq 1000$.

V.8 Extensions

We now extend our results to following two cases:

- Systems where partially accurate information sharing results are valuable to participating parties.
- Systems where each party is selfish in that it considers the information sharing to succeed if the party itself (instead of both parties) obtains the accurate information sharing results.

V.8.1 Systems where Partially Accurate Results are Valuable

In the systems where partially accurate information sharing results are still valuable to participating parties, we cannot use the accuracy measure proposed in the previous part of the chapter. Let V_i^e be the (partially accurate) information sharing result obtained by party P_i. We define an alternative accuracy measure as follows.

$$l_a(s_A, s_D) = \max\left(0, \min_{i\in\{0,1\}}\left(1 - \frac{|V_i^e \setminus (V_0 \cap V_1)| + |(V_0 \cap V_1) \setminus V_i^e|}{|V_0 \cap V_1|}\right)\right). \quad (5.30)$$

As we can see, $l_a(s_A, s_D)$ measures both false positives (i.e., $v \in V_i^e$ but $v \notin V_0 \cap V_1$) and false negatives (i.e., $v \notin V_i^e$ but $v \in V_0 \cap V_1$) of V_i^e compared with $V_0 \cap V_1$. Note that we cannot simply define $l_a(s_A, s_D)$ to be $|V_i^e \cap (V_0 \cap V_1)|/|V_0 \cap V_1|$ because by doing so, an optimal information sharing protocol would be to return V_i as the information sharing result to each party P_i, such that $V_i^e \cap (V_0 \cap V_1) = V_i \cap (V_0 \cap V_1) = V_0 \cap V_1$.

Based on the alternative definition, we can accordingly change the definition of weakly malicious adversaries as follows, in order to make all results in this chapter still valid.

Definition 5.2. An adversary is weakly malicious if and only if the adversary is not semi-honest and has $0 < \sigma < 1/(|V_0 \cap V_1| + 1)$. An adversary is strongly malicious if and only if the adversary is not semi-honest and has $1/(|V_0 \cap V_1| + 1) \leq \sigma \leq 1$.

It is easy to check an above-defined weakly malicious adversary will not intrude privacy if a successful intrusion of privacy always results in at least one of the following two outcomes: 1) the adversary will be convicted as an adversary by the other party, or 2) at least one party cannot obtain $V_0 \cap V_1$. Thus, all results derived in the chapter are still

valid given the alternative accuracy measure and the revised definition of weakly and strongly malicious adversaries.

V.8.2 Systems with Selfish Parties

We now consider systems with parties which consider the information sharing to succeed if a party itself (instead of both parties) obtains the accurate information sharing results. In these systems, we must measure the accuracy of information sharing results individually for each party. In particular, we define the accuracy for party P_i as follows.

$$l_{\mathrm{a}}^{i}(s_{\mathrm{A}}, s_{\mathrm{D}}) = \begin{cases} 1, & \text{if } P_i \text{ obtains } V_0 \bigcap V_1, \\ 0, & \text{otherwise.} \end{cases} \tag{5.31}$$

Based on the new accuracy measure, the objective function of an adversary P_i can be defined accordingly as follows.

$$u_{\mathrm{A}}^{i}(s_{\mathrm{A}}, s_{\mathrm{D}}) = (1-\sigma) l_{\mathrm{a}}^{i}(s_{\mathrm{A}}, s_{\mathrm{D}}) + \sigma l_{\mathrm{p}}(s_{\mathrm{A}}, s_{\mathrm{D}}). \tag{5.32}$$

For such systems, there exists no protocol that can achieve absolute security when a non-semi-honest adversary with $\sigma > 0$ exists in the system. Formally, we have the following theorem.

Theorem 5.10. Given the new accuracy measure, when a non-semi-honest adversary with $\sigma > 0$ exists in the system, there exists no protocol that can simultaneously satisfy the following two conditions

1. If both parties follow the protocol properly without changing their input datasets, at the end of the execution of the protocol, both parties obtain $V_0 \cap V_1$, the size of the dataset of the other party, and nothing else,

2. There is $l_p(s_A, s_D^0) = 0$, where s_A is the optimal attacking method for the adversary.

The proof of this theorem is similar to the proof of Theorem 5.8 and Theorem 5.9. The basic idea is that an adversary can always insert one data point into its input dataset to force an honest party to choose between privacy disclosure and the risk of rejecting another honest party.

Given this theorem, Protocol A, which is designed for weakly malicious adversaries, is no longer effective in these systems. Nonetheless, all results derived for systems with semi-honest adversaries or strongly malicious adversaries, including Protocol B, are still valid.

V.9 Summary

In this chapter, we address issues related to the design of information sharing protocol. Most previous studies investigate the problem and propose solutions based on an assumption that all parties are either honest or semi-honest. This assumption substantially underestimates the capability of adversaries and thus does not always hold in practical situations. We consider a space of more powerful adversaries which include not only semi-honest adversaries but also those who are weakly malicious and strongly malicious. For weakly malicious adversaries, we design an efficient protocol and show that the protocol can preserve privacy effectively. For strongly malicious adversaries, we propose a game theoretic formulation of the system and derive the Nash equilibrium of the game. We evaluate the performance of defensive countermeasure in the Nash

equilibrium and show that with an acceptable loss of accuracy, the privacy of the defending entity can be effectively preserved in many systems.

Again, we remark that in this chapter, we are not promoting specific protocols. Instead, we show that simple and efficient solutions can be developed to deal with malicious adversaries. Specifically, we show simple solutions can be effective if we 1) constrain the adversary goal to be weakly malicious, or 2) allow making a tradeoff between accuracy and privacy.

Many extensions to our work exist, including 1) extending the information sharing function from intersection to other operations, and 2) dealing with multiple parties in the system, including dealing with correlated attacks from multiple adversaries. Besides, our results can be readily applied to various information sharing functions including equijoin and scalar product.

CHAPTER VI

INTEGRATION OF DIFFERENT PROTOCOLS

In this chapter, we study the integration of, and the interaction between, different protocols introduced in previous chapters of this dissertation. In particular, we address the integration of data collection protocol and inference control protocol, and the integration of inference control protocol and information sharing protocol, respectively. We show cases where only one protocol is sufficient, and other cases in which different protocols must be integrated together to accommodate the requirements of the system.

VI.1 Data Collection Protocol and Inference Control Protocol

As we mentioned in Chapter II, the design of data collection protocol and inference control protocol strongly correlate with each other. For example, when the noise insertion approach is used for the data collection protocol, a data-oriented inference control protocol can be applied on the data stored in the data warehouse server. Nonetheless, when our scheme on data collection protocol is used, no data-oriented inference control protocol is needed in the system, because the collected data can be directly used to support data mining. We summarize the relationship between design strategies for data collection protocol and inference control protocol as follows.

- When the original data are stored in the data warehouse server, query-oriented inference control protocol is a practical choice. The reason is that data-oriented protocols have to add noise to the data being mined at run-time. This may result in significant overhead on OLAP query processing.

- When the data stored in the data warehouse are randomized data, data-oriented inference control protocol becomes a natural choice, as the data-oriented protocol can be directly applied to the data warehouse without any run-time randomization of individual data points.
- When the data stored in the data warehouse server are perturbed with our new scheme, no inference control protocol is needed in many systems. This is because only the minimum private information necessary for data mining is included in the (perturbed) data collected by the data warehouse server. Since the perturbed data can be readily used to support data mining, no inference control protocol is needed for systems with moderate privacy requirements. For systems with high levels of privacy concern, query-oriented inference control protocol can be used to further guarantee that no individual data point can be compromised by the data mining servers.

VI.2 Inference Control Protocol and Information Sharing Protocol

Inference control protocol and information sharing protocol are normally transparent to each other, as the inference control protocol enables a data mining server to construct local data mining models, and the information sharing protocol enables a data mining server to share local data mining models and construct global data mining models spanning multiple systems.

Nonetheless, there are certain cases where inference control protocol and information sharing protocol need to be integrated with each other. Recall that in order for the information sharing protocol to work, the participating parties (i.e., data mining servers

from different systems) must have a specifically designed cryptographic algorithm for every data mining task. In cases where such a specific algorithm is unavailable, a possible alternative is for each party to allow other parties (from other systems) to directly access its local data warehouse. In this case, the privacy protection must be implemented in the inference control protocol to accommodate the requirements of information sharing.

The objective of the (new) inference control protocol becomes 1) to allow local data mining servers to learn the minimum private information necessary for data mining, and 2) to prevent remote data mining servers of other systems from inferring private information stored in the data warehouse.

Our cardinality-based inference control protocol can be revised to realize the new inference control protocol. Suppose that each data warehouse server knows the dimension domains (i.e., $d_1 \times \cdots \times d_n$) of the data cubes of all other data warehouses (in other systems). The basic idea of the protocol is for each data warehouse to build a virtual data warehouse. The schema of the virtual data warehouse is a combination of the local (real) data cube and remote data cubes in the other systems. Since the dimension domains of other data warehouses are known, a data warehouse server can still employ the cardinality-based approach to prevent individual data points from being compromised by either local or remote data mining servers.

Although this revised protocol is still efficient and easy to implement, its performance is worse than the original protocol, and may be insufficient for systems that demand high accuracy level. Indeed, since the number of known data points increases dramatically in

the virtual data warehouse (partially due to the fact that we have to conservatively assume that a data mining server from another system has full access to its data warehouse), the data availability level (i.e., percentage of queries answered) of the inference control protocol must decrease considerably. Further study is needed in order to improve the performance of the integration of inference control protocol and information sharing protocol.

CHAPTER VII

SUMMARY AND CONCLUSIONS

VII.1 Summary

In this dissertation, we address the design issues for extracting knowledge from large amounts of data without violating the privacy of data owners. We first introduce an integrated baseline architecture, design principle, and implementation techniques for privacy-preserving data mining systems. Then, we discuss the key components of privacy-preserving data mining systems which are: data collection protocol, inference control protocol, and information sharing protocol. We present and compare strategies for realizing these protocols.

VII.2 Conclusions

We now conclude this dissertation with a discussion on open issues that need to be addressed in order to further improve the performance of privacy-preserving data mining techniques.

- Heterogeneous Privacy Requirements: The design of privacy-preserving data mining techniques depends on the specification of privacy protection levels required by the data owners. Most existing studies assume (at least partially) homogenous privacy requirements. That is, all data owners have the same level of privacy requirement on all of their data and/or all attributes. This assumption simplifies the design and implementation of privacy-preserving data mining techniques, but cannot reflect the privacy concerns in practice. Indeed, due to multiple survey results [14], [33], [35], [59], different people have diversified

privacy requirements on different data and/or different attributes. The work in this dissertation (e.g., our scheme for data collection protocol) removes part of the assumption, as we allow different data providers to specify different levels of privacy protection on their data. Nonetheless, we still cannot allow a data provider to explicitly assign different privacy protection levels on different attributes. It would be challenging, but potentially beneficial, to design and implement new techniques that fully address the heterogeneous privacy requirements of data owners.

- Integration of Different Protocols: As we mentioned in Chapter VI, an effective integration of the three protocols is needed in many real systems. Nonetheless, the integration of different protocols has not received enough attention from the research community, partially due to the fact that these three protocols are traditionally studied in different fields as separate problems. This dissertation proposes an integrated architecture of privacy-preserving data mining systems that can serve as a platform for the integration of these protocols. Further studies are needed to enable such integration in an effective and efficient manner.
- Privacy-Preserving Anomaly Mining: Data mining has been extensively used to detect anomalies in datasets (e.g., intrusion detection based on log files). Nonetheless, in the research community, privacy protection in detecting anomalies (i.e., diamond mining) [52] has not received the same level of attention as privacy protection in constructing predictive models (i.e., coal mining). Since anomaly detection is an important application of data mining,

and is significantly important to many disciplines (e.g., information security, biology, finance), it would be necessary and beneficial to thoroughly investigate issues related to the design of privacy-preserving data mining techniques for anomaly detection.

REFERENCES

[1] D. Agrawal and C. C. Aggarwal, "On the Design and Quantification of Privacy Preserving Data Mining Algorithms," *Proc. 20th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, pp. 247-255, 2001.

[2] R. Agrawal, D. Asonov, and R. Srikant, "Enabling Sovereign Information Sharing Using Web Services," *Proc. 23rd ACM SIGMOD International Conference on Management of Data*, pp. 873-877, 2004.

[3] R. Agrawal, A. Evfimievski, and R. Srikant, "Information Sharing Across Private Databases," *Proc. 22nd ACM SIGMOD International Conference on Management of Data*, pp. 86-97, 2003.

[4] R. Agrawal and R. Srikant, "Privacy-Preserving Data Mining," *Proc. 19th ACM SIGMOD International Conference on Management of Data*, pp. 439-450, 2000.

[5] R. Agrawal, R. Srikant, and D. Thomas, "Privacy Preserving OLAP," *Proc. 25th ACM SIGMOD International Conference on Management of Data*, pp. 251 - 262, 2005.

[6] S. Agrawal and J. Haritsa, "A Framework for High-Accuracy Privacy-Preserving Mining," *Proc. 21st International Conference on Data Engineering*, pp. 193-204, 2005.

[7] S. Agrawal, V. Krishnan, and J. Haritsa, "On Addressing Efficiency Concerns in Privacy-Preserving Mining," *Proc. Ninth International Conference on Database Systems for Advanced Applications*, pp. 113-124, 2004.

[8] R. A. Baeza-Yates and B. Ribeiro-Neto, *Modern Information Retrieval*. New York, NY: Addison-Wesley, 1999.

[9] R. J. Bayardo and R. Agrawal, "Data Privacy through Optimal *k*-Anonymization," *Proc. 21st International Conference on Data Engineering*, pp. 217-228, 2005.

[10] C. Blake and C. Merz, *UCI repository of machine learning databases*. Irvine, CA: University of California, Department of Information and Computer Science, 1998.

[11] M. Blum, "How to Exchange (secret) Keys," *ACM Transactions on Computer Systems*, vol. 1, pp. 175-193, 1983.

[12] R. Conway and D. Strip, "Selective Partial Access to a Database," *Proc. ACM/CSC-ER Annual Conference*, pp. 85-89, 1976.

[13] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. New York, NY: Wiley-Interscience, 1991.

[14] L. F. Cranor, J. Reagle, and M. S. Ackerman, "Beyond Concern: Understanding Net Users' Attitudes About Online Privacy," AT&T Labs-Research TR 99.4.3, 1999.

[15] J. Domingo-Ferrer, *Inference Control in Statistical Databases*. New York, NY: Springer, 2002.

[16] W. Du, Y. S. Han, and S. Chen, "Privacy-Preserving Multivariate Statistical Analysis: Linear Regression and Classification," *Proc. Fourth SIAM International Conference on Data Mining*, pp. 222-233, 2004.

[17] W. Du and H. Polat, "Privacy-Preserving Collaborative Filtering Using Randomized Perturbation Techniques," *Proc. Third IEEE International Conference on Data Mining*, pp. 625-628, 2003.

[18] W. Du and Z. Zhan, "Building Decision Tree Classifier on Private Data," *Proc. IEEE International Conference on Privacy, Security and Data Mining*, pp. 1-8, 2002.

[19] W. Du and Z. Zhan, "Using Randomized Response Techniques for Privacy-Preserving Data Mining," *Proc. 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 505-510, 2003.

[20] T. ElGamal, "A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms," *IEEE Transactions on Information Theory*, vol. 31, pp. 469-472, 1985.

[21] European Union, "Directive 95/46/EC on the Protection of Individuals with Regard to the Processing of Personal Data and on the Free Movement of Such Data," Brussels, Belgium, 1998.

[22] S. Even, O. Goldreich, and A. Lempel, "A Randomizing Protocol for Signing Contracts," *Communications of the ACM*, vol. 28, pp. 637-647, 1985.

[23] A. Evfimievski, J. Gehrke, and R. Srikant, "Limiting Privacy Breaches in Privacy Preserving Data Mining," *Proc. 22nd ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pp. 211-222, 2003.

[24] A. Evfimievski, R. Srikant, R. Agrawal, and J. Gehrke, "Privacy Preserving Mining of Association Rules," *Proc. 8th ACM SIGKDD International Conference on Knowledge Discovery in Databases and Data Mining*, pp. 217-228, 2002.

[25] M. J. Freedman, K. Nissim, and B. Pinkas, "Efficient Private Matching and Set Intersection," *Advances in Cryptography: Proc. Eurocrypt 2004*, pp. 1-19, 2004.

[26] D. Fudenberg and J. Tirole, *Game Theory*: Boston, MA: MIT Press, 1991.

[27] O. Goldreich, *The Foundations of Cryptography*, vol. 2. Cambridge, United Kingdom: Cambridge University Press, 2004.

[28] G. H. Golub and C. F. V. Loan, *Matrix Computation*. Baltimore, MD: John Hopkins University Press, 1996.

[29] J. Han and M. Kamber, *Data Mining Concepts and Techniques*. San Francisco, CA: Morgan Kaufmann, 2001.

[30] Department of Health and Human Services, "Health Insurance Portability and Accountability Act," Washington, DC, 2002.

[31] Z. Huang, W. Du, and B. Chen, "Deriving Private Information from Randomized Data," *Proc. 24th ACM SIGMOD International Conference on Management of Data*, pp. 37 - 48, 2005.

[32] B. A. Huberman, M. Franklin, and T. Hogg, "Enhancing Privacy and Trust in Electronic Communities," *Proc. 1st ACM Conference on Electronic Commerce*, pp. 78-86, 1999.

[33] IBM, "The IBM-Harris Multi-national Customer Privacy Survey," New York, NY, 1999.

[34] IBM, "Privacy-Preserving Data Mining Project," June 10, 2006, http://www.zurich.ibm.com/pri/projects/datamining.html.

[35] Japanese Ministry of Posts and Telecommunications, "Ministry of Posts and Telecommunications Survey," Tokyo, Japan, 1999.

[36] M. Kantarcioglu and C. Clifton, "Privacy-preserving Distributed Mining of Association Rules on Horizontally Partitioned Data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 16, pp. 1026-1037, 2004.

[37] M. Kantarcioglu and J. Vaidya, "Privacy Preserving Naive Bayes Classifier for Horizontally Partitioned Data," *Workshop on Privacy Preserving Data Mining held in association with the Third IEEE International Conference on Data Mining*, 2003.

[38] H. Kargupta, S. Datta, Q. Wang, and K. Sivakumar, "On the Privacy Preserving Properties of Random Data Perturbation Techniques," *Proc. Third IEEE International Conference on Data Mining*, pp. 99-106, 2003.

[39] L. Kissner and D. Song, "Privacy-Preserving Set Operations," *Proc. 25th Annual International Cryptology Conference*, pp. 241-257, 2005.

[40] M. Li and P. Vitanyi, *An Introduction to Kolmogorov Complexity and Its Applications*. New York, NY: Springer Verlag, 1997.

[41] Y. Lindell and B. Pinkas, "Privacy Preserving Data Mining," *Proc. 12th Annual International Cryptology Conference on Advances in Cryptology*, pp. 36-54, 2000.

[42] M. Luby, S. Micali, and C. Rackoff, "How to Simultaneously Exchange a Secret Bit by Flipping a Symmetrically-Biased Coin," *Proc. 24th Annual Symposium on the Foundations of Computer Science*, pp. 11-12, 1983.

[43] D. McCullagh, "Database Nation: The Upside of Zero Privacy," *Reason*, vol. 6, pp. 26-35, 2004.

[44] P. Paillier, "Public-key Cryptosystems Based on Composite Degree Residuosity Classes," *Advances in Cryptology - EUROCRYPT 99*, pp. 223-238, 1999.

[45] S. C. Pohlig and M. E. Hellman, "An Improved Algorithm for Computing Logarithms in GF(p) and Its Cryptographic Significance," *IEEE Transactions on Information Theory*, vol. 24, pp. 106-111, 1978.

[46] M. O. Rabin, "How to Exchange Secrets by Oblivious Transfer," Technical Memo TR-81, Aiken Computer Laboratory, Harvard University, 1981.

[47] S. Rizvi and J. Haritsa, "Maintaining Data Privacy in Association Rule Mining," *Proc. 28th International Conference on Very Large Data Bases*, pp. 682-693, 2002.

[48] F. Schoeman, "Philosophical Dimensions of Privacy: An Anthology." Cambridge: Cambridge University Press, 1984.

[49] A. Shamir, R. L. Rivest, and L. M. Adleman, "Mental Poker," in *Mathematical Gardner*. Belmont, California: Wadsworth, 1981, pp. 37-43.

[50] A. C. Tamhane and D. D. Dunlop, *Statistics and Data Analysis: From Elementary to Intermediate*. Englewood Cliffs, NJ: Prentice-Hall, 2000.

[51] J. F. Traub, Y. Yemini, and H. Wozniakowski, "The Statistical Security of a Statistical Database," *ACM Transactions on Database Systems*, vol. 9, pp. 672-679, 1984.

[52] J. Vaidya and C. Clifton, "Privacy-Preserving Outlier Detection " *Proc. Fourth IEEE International Conference on Data Mining*, pp. 233-240, 2004.

[53] J. Vaidya and C. Clifton, "Privacy-Preserving Top-K Queries," *Proc. 21st International Conference on Data Engineering*, pp. 545-546, 2005.

[54] J. Vaidya and C. Clifton, "Privacy Preserving Association Rule Mining in Vertically Partitioned Data," *Proc. 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 639-644, 2002.

[55] J. Vaidya and C. Clifton, "Privacy preserving Naive Bayes Classifier for Vertically Partitioned Data," *Proc. Fourth SIAM Conference on Data mining*, pp. 330-334, 2004.

[56] V. S. Verykios, E. Bertino, I. N. Fovino, L. P. Provenza, Y. Saygin, and Y. Theodoridis, "State-of-the-art in Privacy Preserving Data Mining," *SIGMOD Record*, vol. 33, pp. 50-57, 2004.

[57] L. Wang, S. Jajodia, and D. Wijesekera, "Securing OLAP Data Cubes Against Privacy Breaches," *Proc. 25th IEEE Symposium on Security and Privacy*, pp. 161-175, 2004.

[58] L. Wang, D. Wijesekera, and S. Jajodia, "Cardinality-Based Inference Control in Data Cubes," *Journal of Computer Security*, vol. 12, pp. 655 - 692, 2004.

[59] A. F. Westin, "Consumers, Privacy, and Survey Research," New York, NY, Council of American Survey Research Organizations Annual Conference, 2003.

[60] A. F. Westin, *Privacy and Freedom*. New York, NY: Atheneum, 1967.

[61] R. N. Wright and Z. Yang, "Privacy-preserving Bayesian Network Structure Computation on Distributed Heterogeneous Data," *Proc. 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 713-718, 2004.

[62] Z. Yang, S. Zhong, and R. N. Wright, "Privacy-Preserving Classification of Customer Data without Loss of Accuracy," *Proc. Fifth SIAM International Conference on Data Mining*, pp. 92-102, 2005.

[63] A. C. Yao, "Protocols for Secure Computations (Extended Abstract)," *23rd Annual Symposium on Foundations of Computer Science*, pp. 160--164, 1982.

[64] Z. Zheng, R. Kohavi, and L. Mason, "Real World Performance of Association Rule Algorithms," *Proc. Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 401-406, 2001.

VITA

Nan Zhang received his Bachelor of Science degree in Computer Science from Beijing University in July, 2001. He entered the Department of Computer Science at Texas A&M University in September, 2001, and received his Doctor of Philosophy degree from Texas A&M University in December, 2006. His research interests include information security and privacy, database and data mining, and distributed systems.

Nan Zhang may be reached at the Department of Computer Science, Texas A&M University, College Station, TX 77840-3112, USA. His email address is nzhang@tamu.edu.