Towards Privacy-Preserving Integration of Distributed Heterogeneous Data

Pawel Jurczyk
Department of Math&CS
Emory University
Atlanta, GA, USA
pjurczy@emory.edu

Li Xiong
Department of Math&CS
Emory University
Atlanta, GA, USA
lxiong@emory.edu

ABSTRACT

More and more applications rely heavily on large amounts of data in the distributed storages collected over time or produced by large scale scientific experiments or simulations. An important fact is that many organizations collect, store, and use various types of information about individuals. In consequence, such data sharing is subject to constraints imposed by privacy of individuals or data subjects as well as data confidentiality of institutions or data providers. Given a query spanning multiple databases, it should be executed transparently and efficiently. And most importantly, the results should not contain individually identifiable information and institutions should not reveal their databases to each other apart from the query results. In this paper, we propose a distributed anonymization protocol that allows independent data providers to build a virtual anonymized database from horizontally partitioned databases, and a secure query protocol that allows clients to query those virtual databases. We also propose a distributed data sharing and integration architecture for querying these distributed heterogeneous and possibly private databases. Our system provides efficient and scalable privacy-preserving query execution interface that integrates data seamlessly and transparently.

Categories and Subject Descriptors

H.2 [**DATABASE MANAGEMENT**]: General, Systems

General Terms

Algorithms, Experimentation, Design, Security

1. INTRODUCTION

Current information technology enables large amounts of data to be collected in the distributed storages from business transactions, large scale scientific experiments, or monitoring devices. Often such data contain personal information about individuals, such as customers and patients. Government and organizations increasingly recognize the critical value and opportunities in sharing such a wealth of information across multiple distributed, private, and possibly untrusted databases.

Consider a system that integrates the air and rail transportation networks with patients databases and demographic databases in order to model the large scale spread of infectious diseases (such as the SARS epidemic or pandemic influenza). Rail and air transportation databases are distributed among hundreds of local servers, demographic information is provided by a few global database servers and patient data is provided by groups of cooperating hospitals.

Another example is the Shared Pathology Informatics Network (SPIN)[1] initiative by the National Cancer Institute that attempts to provide a system offering search interfaces for existing electronic databases at institutions across the country to locate human specimens and associated clinical and pathologic data needed for cancer research. The goal of the system is to enable investigators to query archived information in multiple institutions transparently.

While there are increasing needs for sharing data that contain personal information across distributed, heterogeneous and possibly private data sources, it remains a challenge to ensure security, privacy, and interoperability for such data sharing. One key issue is that the data sharing is subject to two constraints: 1) privacy of individuals or data subjects, and 2) data confidentiality of institutions or data providers. For example, personal health information is protected under the Health Insurance Portability and Accountability Act (HIPAA)[2][3] and cannot be revealed without de-identification or anonymization. In addition, institutions may not want to reveal their private databases to each other apart from the final computation result for various reasons. Ideally, given a query spanning multiple databases, query results should not contain individually identifiable information. In addition, institutions should not reveal their databases to each other apart from the query results.

Aside from the privacy and security requirements, the class of applications mentioned in above scenarios is characterized by a number of features and requirements. First, the *heterogenity* of data sources requires a unified and seamless

PIKM'08, October 30, 2008, Napa Valley, California, USA.

[1]Shared Pathology Informatics Network. http://www.cancerdiagnosis.nci.nih.gov/spin/
[2]Health Insurance Portability and Accountability Act (HIPAA). http://www.hhs.gov/ocr/hipaa/.
[3]State law or institutional policy may differ from the HIPAA standard and should be considered as well.

data representation and query interface for the applications. The data sources can be structurally or semantically different. In addition, they may have different privacy requirements in that some data sources may contain personal data that require anonymization while others may not. When sensitive information is being queried, it needs to be recognized automatically and transparently from users point of view and special mechanisms have to be deployed to guarantee security and privacy of the data. Second, the scale of the applications can vary from a handful to several hundreds of nodes and requires good *scalability* as well as data query and update functionalities with *data consistency*. Lastly, the *dynamics* of resource and network conditions requires the distributed data sharing applications to adapt dynamically in both query processing and transaction management in order to achieve scalability and data consistency.

Contributions. In this paper, we propose a framework for privacy-preserving integration of distributed heterogeneous databases that are horizontally partitioned. We mainly focus on architectural issues. We also present a sketch of distributed anonymization approach to address the privacy and security issue and a scalable and secure distributed framework for querying the distributed and heterogeneous data sources. Our research consists of two main contributions.

First, to enable anonymization of distributed data, we propose a *distributed k-anonymization protocol* that allows multiple data providers with horizontally partitioned databases to build a virtual k-anonymized database based on the integration (or union) of the data, and a *distributed secure querying protocol* that allows clients to query the virtual anonymized database. As the output of the anonymization protocol, each database produces a local anonymized dataset and their union forms a virtual database that is guaranteed to be k-anonymous. When users query the virtual database, each individual database executes the query on its local anonymized dataset, and then engage in the distributed querying protocol to assemble the results that are guaranteed to be k-anonymous. Both protocols utilize multi-party protocols for sub-operations such that information disclosure between individual databases is minimal.

Second, we propose a novel architecture that enables querying of the distributed, heterogeneous and possibly private databases. Public and private databases coexist in the system and form a virtual database. Clients do not interact with one given database at any time, but rather query the virtual system as whole. Our query architecture provides data services for querying and operating data from heterogeneous data sources, and the mechanisms for distributed anonymization and secure query execution are integrated with the system and can be used transparently from users' point of view. The proposed architecture provides privacy-preserving data sharing framework that is scalable, supports heterogeneous data sources, and responds to dynamic network and resources conditions.

Organization. The remainder of this paper is organized as follows. The next section presents background information on secure data sharing, secure multi-party computation, basic approaches to distributed data integration and querying, and introduces a high-level architecture for our proposed data sharing framework. Section 3 discusses the privacy model we are using. Section 4 presents our distributed anonymization protocol and distributed secure querying protocol. Section 5 presents general architecture for distributed querying architecture, DObjects, and its extension for sharing private data. Finally, Section 6 concludes the paper and gives idea of future research directions.

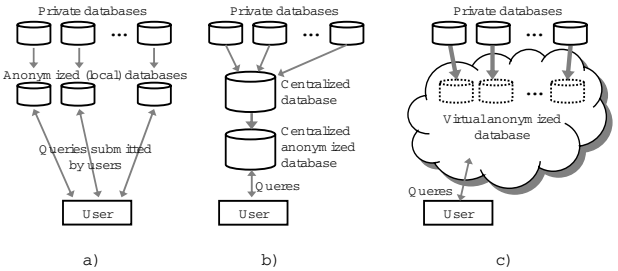

Figure 1: Possible architectures for privacy preserving distributed data publishing.

2. BACKGROUND AND OVERVIEW

In this section we provide some background information on privacy-preserving data sharing and integration of distributed heterogeneous data. We then give an overview of our proposed architecture for sharing and querying distributed heterogeneous and possibly private databases.

Privacy preserving data publishing. Privacy preserving data publishing for a single database has been extensively studied in recent years. A large body of work contributes to data anonymization that transforms a dataset to meet a privacy principle such as k-anonymity using techniques such as generalization, suppression (removal), permutation and swapping of certain data values so that it does not contain individually identifiable information [9, 23, 18, 2, 1, 5, 3, 26, 15, 16, 17, 22, 13, 24, 25].

There are a number of potential approaches one may apply to enable privacy preserving data publishing for distributed databases. A naive approach is for each data custodian to perform data anonymization independently as shown in Fig. 1a. Data recipients or clients can then query the individual anonymized databases or an integrated view of them. One main drawback of this approach is that data is anonymized before the integration and hence will cause the data utility to suffer. In addition, individual databases reveal their ownership of the anonymized data.

An alternative approach assumes an existence of third party that can be trusted by each of the data owners as shown in Fig. 1b. In this scenario, data owners send their data to the trusted third party where data integration and anonymization are performed. Then, clients can query the centralized database. However, finding such a trusted third party is not always feasible. Compromise of the server by hackers could lead to a complete privacy loss for all participating parties.

In this paper, we propose a distributed data anonymization approach as illustrated in Fig. 1c. In this approach, data owners participate in distributed protocols to produce a *virtual* integrated and anonymized database which can be then queried by clients. Important to note is that the anonymized data still resides at individual databases and the integration and anonymization of the data is performed through the distributed protocols.

This approach has its roots in the secure multi-party com-

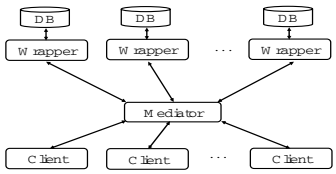

Figure 2: Typical mediator-based architecture.

putation (MPC) problem [7, 6]. In MPC, a given number of participants, each having a private data, wants to compute the value of a public function. A MPC protocol is secure if no participant can learn more from the description of the public function and the result of function. While there are general secure MPC protocols, they require substantial computation and communication costs and are impractical for large multi-party database problems. By exploiting the inherent anonymity of multiple parties and probabilistic methods, our approach provides a desired tradeoff of absolute security for efficiency compared to traditional MPC approaches.

There are also some works focused on data anonymization of distributed databases. A two-party framework along with an application that generates k-anonymous data from two vertically partitioned sources without disclosing data from one site to the other was presented in [10]. Provably private solutions for k-anonymization in the distributed scenario by maintaining end-to-end privacy from the original customer data to the final k-anonymous results was discussed in [27]. In contrast, our work is aimed at horizontal data distribution and arbitrary number of sites. In addition, our protocols are not based on heavy cryptographic primitives. We exploit inherent anonymity of large number of participating sites and utilize probabilistic methods to achieve minimal information disclosure and minimal overhead.

Integration of distributed heterogeneous data. The distributed data anonymization techniques introduced above require a query execution interface that provides scalable and secure access to heterogeneous and distributed data sources.

At the first glance, distributed database systems have been extensively studied and many systems have been proposed over the years. Earlier distributed database systems [14], such as R* and SDD-1, share modest targets for network scalability (a handful of distributed sites) and assume homogeneous databases. The focus is on encapsulating distribution with ACID guarantees. Later distributed database or middleware systems, such as Garlic [4] or DISCO [20] target large-scale heterogeneous data sources. Many of them employ a *centralized* mediator-wrapper based architecture (see Figure 2) to address the database heterogeneity in the sense that a single mediator server integrates distributed data sources through wrappers. The query optimization focuses on integrating wrapper statistics with traditional cost-based query optimization for single queries spanning multiple data sources. As the query load increases, the centralized mediator may become a bottleneck. Most recently, Internet scale query systems, such as Astrolabe [21] and PIER [8], target thousands or millions of massively distributed homogeneous data sources with a peer-to-peer (P2P) or hierarchical network architecture and focus on efficient query routing schemes for network scalability. However they sacrifice on

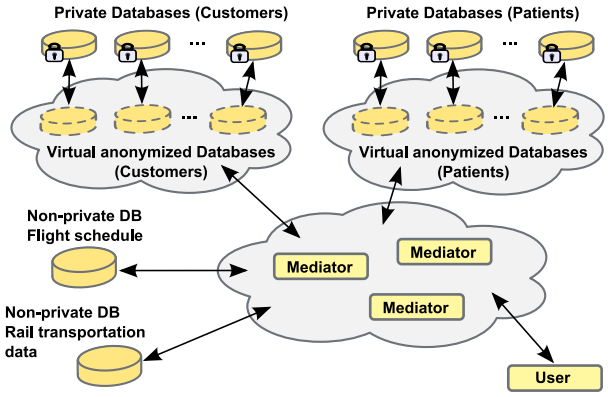

Figure 3: General architecture for secure data sharing.

functionalities of complex queries and data updates and typically relax the consistency guarantee.

Considering above issues, we designed a general-purpose query infrastructure called DObjects [12] that is based on a *distributed* mediator-wrapper based architecture. The system includes a distributed query processing engine that deploys and executes (sub)queries on system nodes in a *dynamic* (based on nodes' on-going knowledge of the data sources, network and node conditions) and *iterative* (right before the execution of each query operator) manner to optimize both response time and throughput. While it is not the aim of DObjects to be superior to any of the above works, the system distinguishes itself by addressing an important problem space that has been overlooked, namely, integrating large-scale heterogeneous data sources with both network and query load scalability as well as transaction semantics. In spirit, DObjects is a *distributed* mediator-based system in which a federation of mediators and wrappers forms a virtual system in a P2P fashion.

Proposed architecture for privacy preserving integration of distributed heterogeneous data. In this paper we propose a novel approach to scalable and secure querying of distributed, heterogeneous and possibly private databases. The main idea is illustrated in Figure 3. We assume that public and private databases coexist in the system and form a virtual system. Clients connecting to the system do not interact with one given database at any time, but rather query the virtual system that exploits a distributed mediator-wrapper architecture offered by the DObjects framework. It is the role of the system to parse, optimize and deploy user's queries, integrate data, and return results. We assume that data owners with private information participate in distributed protocols to produce a *virtual* integrated and anonymized database which can be then used in the system. Important to note is that the anonymized data still resides at individual databases and the integration and anonymization of the data is performed through the distributed protocols.

3. PRIVACY MODEL

In this section we present the privacy goals that we focus on in the paper, followed by privacy models for characterizing how these privacy goals are achieved. As we identified in Section 1, given a query spanning multiple databases, our

privacy goals are two fold. First, the query results should not contain individually identifiable information. Second, individual databases should not reveal their data to each other apart from the query results. We describe the models we use for each of these two goals.

Individual identifiability. Among the many privacy principles based on individual identifiability, k-anonymity [19] is the most widely accepted and serves as the basis for many others, and hence, will be used in our current approach. The general approach and protocol structure, however, is orthogonal to these privacy principles and it is on our research agenda to incorporate more advanced privacy principles into our protocol set.

In defining anonymization, attributes of a given relational table T, are characterized into three types. *Unique identifiers* are attributes that identify individuals. Known identifiers are typically removed entirely from released microdata. *Quasi-identifier set* is a minimal set of attributes (X_1, ..., X_d) that can be joined with external information to re-identify individual records. *Sensitive attributes* are attributes that should be prevented from being associated with a unique entity by an adversary.

The k-anonymity model provides an intuitive requirement for privacy in stipulating that no individual record should be uniquely identifiable from a group of k with respect to the quasi-identifier set. The set of all tuples in T containing identical values for the quasi-identifier set X_1, ..., X_d is referred to as an *Equivalence Class*. T is *k-anonymous* with respect to X_1, ..., X_d if every tuple is in an equivalence class of size at least k. A *k-anonymization* of T is a transformation or generalization of the data T such that the transformation is k-anonymous.

Data confidentiality. Our second privacy goal follows the goal of secure MPC but is less stringent. Instead of attempting to guarantee absolute security in which individual databases reveal nothing about their data to each other apart from the query results, we wish to minimize data exposure among the multiple parties. We also adopt the *semi-honest* model commonly used in secure MPC problems. A semi-honest party follows the rules of the protocol, but it can attempt to learn additional information about other nodes by analyzing the data received during the execution of the protocol.

4. DISTRIBUTED ANONYMIZATION

In this section we describe our distributed anonymization approach. We first describe the general protocol structure, then present the distributed anonymization protocol, followed by the distributed querying protocol.

We assume that the data are split horizontally among n sites ($n > 2$) and each site owns a private database d_i. In addition, the *quasi-identifier* of each local database is uniform among all the sites (the assumption of uniform quasi-identifier is plausible for horizontally split data). The sites engage in a distributed anonymization protocol where each site produces a local anonymized dataset a_i and their union forms a virtual database that is guaranteed to be k-anonymous. Note that a_i is not required to be k-anonymous by itself. When users query the virtual database, each individual database executes the query on a_i and then engage in a distributed querying protocol to assemble the results that are guaranteed to be k-anonymous.

4.1 Protocol Structure

The proposed protocols are designed to run over a decentralized network. The protocol structure is presented in Figure 4. Nodes are mapped to a ring topology randomly. We assume that each node knows its predecessor and successor. Each node has a local computation module that executes its part of the protocol independently and passes the computation result along the ring.

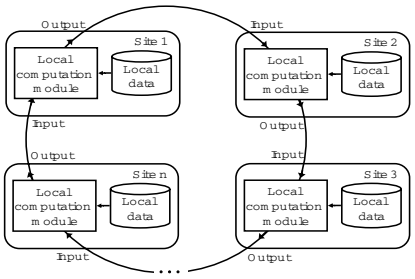

Figure 4: Protocol structure.

Algorithm 1 Distributed anonymization algorithm - leading site ($i = 0$)

1: **function** split(set d_0)
2: Compute range of values for each attribute from quasi-identifier in set $d = \bigcup d_i$ (using secure min/max protocol)
3: Choose best split attribute a with largest range of values
4: Find median value m of a in set $d = \bigcup d_i$ (using secure median protocol)
5: Send a and m to node 1
6: Split set d_0, create two sets, s_0 containing items *smaller than* m and g_0 containing items *greater than* m. Distribute median items among s_i and g_i.
7: Find $size_{left} = |\bigcup s_i|$ and $size_{right} = |\bigcup g_i|$ (using secure sum protocol)
8: **if** $size_{left} > 2 * k$ **then**
9: Send $split\_left = true$ to node 1
10: **call** split(s_0)
11: **else**
12: Send $split\_left = false$ to node 1
13: **end if**
14: **if** $size_{right} > 2 * k$ **then**
15: Send $split\_right = true$ to node 1
16: **call** split(g_0)
17: **else**
18: Send $split\_right = false$ to node 1
19: **end if**
20: **end function** split

4.2 Distributed Anonymization Protocol

Our distributed algorithm for anonymization is based on the Mondrian algorithm [16] that uses greedy recursive partitioning of the (multidimensional) quasi-identifer domain space. It recursively chooses the split attribute with the largest normalized range of values, and (for continuous or ordinal attributes) partitions the data around the median value of the split attribute. This process is repeated until no allowable split remains, meaning that a particular region cannot be further divided without violating the anonymity

constraint, or constraints imposed by value generalization hierarchies.

Algorithm 2 Distributed anonymization algorithm - non-leading node ($i > 0$)

```
1: function split(set d_i)
2: Read split attribute a and median value m from node (i − 1) and pass them to node i + 1
3: Split set d_i into s_i containing items smaller than m and g_i containing items greater than m. Distribute median items among s_i and g_i.
4: Read split_left from node i − 1 and pass it to node i + 1
5: if split_left then
6:    call split(s_i)
7: end if
8: Read split_right from node i − 1
9: Send split_right to node i + 1
10: if split_right then
11:    call split(g_i)
12: end if
13: end function split
```

The key idea for the distributed anonymization protocol is to use a set of secure atomic multi-party protocols to realize the Mondrian method for the distributed setting. We assume a leading site is selected for the protocol. The main protocol for the leading and other sites are presented in Algorithm 1 and 2 respectively. The steps performed at the leading site are similar to the centralized Mondrian method. It chooses the best attribute (with largest spread) for split. Next, it performs the split and recursively checks whether further split of new subsets is possible. When selecting the best split attribute, the leading site needs to have the knowledge of the ranges of values of each attribute with respect to data items located at all sites. So a secure min/max protocol is used to compute the minimum and maximum value of each attribute across the databases. The leading site can then compute the range of each attribute and select the best attribute with largest range. When finding the split point for partitioning, a secure median protocol is used to find the median value of the attribute with respect to the data across the databases. Finally, when determining whether a partition can be further split, a secure sum protocol is used to count the total number of tuples of the partition across the databases. The details of secure min/max, sum and median algorithms mentioned above can be found in [11].

4.3 Secure Query Protocol

The distributed anonymization protocol enables set of nodes to produce a virtual k-anonymous database based on the union of the data horizontally split among the nodes. At the end of the protocol, the local anonymized datasets are not necessary k-anonymized. However, the union of datasets forms the virtual database and is guaranteed to satisfy k-anonymity requirement. Our approach also includes a distributed querying protocol that allows users to query this virtual database. When a query is received, each database runs the query against its local randomized dataset and the results are then unioned and included in the response. As we require that ownership of items is not revealed during the querying phase, we propose a novel and efficient secure set union protocol for this purpose. Note that the distributed anonymization protocol has to be only run for first query that is submitted to the system. All further queries can (and should) use results of the initial anonymization. Low overhead of the union protocol we propose below guarantees reasonable query response times.

Given a set of nodes with private data items, the problem is to compute the union of data items while minimizing data disclosure of the nodes to each other besides the final result. Our privacy goal is to prevent an adversary from being able to determine the owner of items from the final result. Formally, given n sites, and each site holding a local set of data tuples or items x_i, we wish to compute $X = \bigcup x_i$ while minimizing the probability of a node revealing x_i to other nodes. Given our usage for the distributed querying protocol, we preserve duplicate items. In other words, if the same data item appears more than once in local subsets, all the same items will appear in the final result. The protocol can be easily modified to remove duplicates if necessary.

Naive approach. A naive way to compute the union of data items from distributed nodes without a central server is to have the nodes pass their data items along a ring. The first node sends its items to the second node. The second node adds its own items to the intermediate result from the first node and sends it to the third node and so on. The union is found at the end of the round.

Clearly, the protocol does not offer good data privacy. First, the starting node has *provable exposure* to its successor regarding its data items. Second, the nodes that are close to the starting node in the ring have a fairly high probability disclosing their data items.

Secure set union protocol. The *secure set union protocol* utilizes randomization to minimize data exposure. There are two key ideas to the protocol. The first idea is to introduce random items by the starting node so that it will not suffer from provable exposure. The second idea is to randomly select a starting node so that nodes close to the starting node on the ring will not suffer from a high probability of data disclosure.

Algorithm 3 Secure set union protocol.

```
1: INPUT: x_i: local subset contributing to union
2: Choose random t_i
3: t_max ← max(t_1..t_n) (using secure max protocol)
4: if t_i = t_max then
5:    Generate set of random items r
6:    Send r ∪ x_i to successor
7:    Receive X from predecessor
8:    Result ← (X − r)
9: else
10:   Receive X from predecessor
11:   Send X ∪ x_i to successor
12: end if
```

The protocol works as follows. In the initialization or leader selection round, each node generates a random number t_i from predefined range. The node with highest value of t is elected to be the leader node. The secure max protocol can be used to find the highest value of t. In the main protocol round, the leader node i generates a random set r and adds its local subset x_i to this random set. Then it passes its intermediate result to the node $i+1$. Starting from this point, each node j adds its local subset x_j to the intermediate result and passes the result to node $j + 1$. When node i receives the result from its predecessor, the set union can be found by removing random items r from this set. A sketch of the algorithm is presented in Algorithm 3. For a full analysis of this approach we refer readers to [11].

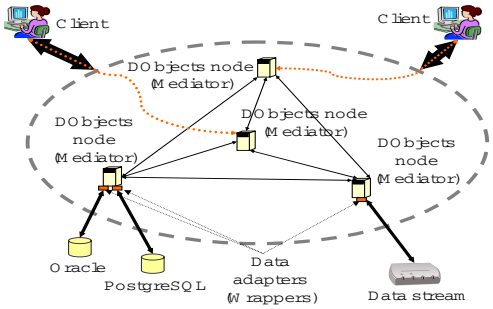

Figure 5: DObjects system architecture.

An important issue is to generate a good random set r. Leading site should be capable of generating items that look legitimate to other nodes and are indistinguishable from real data. The first issue is a domain of attributes. For numeric attributes the leading site can use secure min/max algorithm to obtain a range of valid values. On the other hand for attributes with closed set of values (names or geographic entities) well-known dictionaries can be exploited. Aside from the domain determination, two additional issues play critical role: the distribution of values and correlation between attributes. Most of attributes, such as age or weight, can be expected to follow normal distribution and thus can be generated using such distribution. Credible data generation also requires taking into account correlation between attributes (one can expect that the weight and height values are strongly correlated). Therefore, before generation of random items the leading site should perform correlation analysis first, and then generate dependent attributes based on the depended values.

5. DATA SHARING ARCHITECTURE

In this section we present an overview of DObjects, a distributed mediator-based framework that serves as an underlying data integration and query layer for the complete architecture for privacy-preserving data sharing. We also present a vision of the complete framework that integrates distributed anonymization and secure query protocols with the DObjects framework.

5.1 DObjects Data Services

Figure 5 presents our vision of deployed DObjects framework. The system has no centralized services and uses the *metacomputing* paradigm as a resource sharing substrate. Each node in the system serves as a *mediator* and provides its *computational power* that can be used by others during query execution. Nodes can also serve as a data adapter or *data wrapper* that can pull data from external data sources and transform it to a uniform format that is expected while building query responses. Users can connect to any system node; however, while the physical connection is established between a client and one of the system nodes, the logical connection is established between a client node and a virtual system consisting of all available nodes.

Data model. DObjects uses *persistent entities* as its data model which are data represented as objects. From user's perspective, query responses are objects of desired type. Each data object has a set of *attributes*, divided into two groups: *simple* and *referential*. Simple attributes represent simple types, such as numbers or strings. Referential attributes follow an object-oriented idiom and allow the definition of *association*, *composition* or *collection* relations between data objects. Thus, when a referential attribute is accessed, another persistent entity, or a collection of persistent entities, is obtained.

A set of available data types in the system along with their attributes is defined in the system configuration. Each configuration entry has a full description of an object, i.e. its type name and a list of simple and referential attributes. When a referential attribute is defined, one has to specify the foreign key information that is required to join the referencing object and referenced object. It also specifies a list of nodes (sources) where given objects can be found. Each source is specified with: 1) name of the node, 2) remote data object name, and 3) attribute mappings that define the semantic mappings between the remote data object and the current object. There is no centralized copy of the global configuration. For systems with a handful DObjects nodes (the number of data sources can be still large), the configuration can be replicated and synchronized at every node as the cost of synchronization will be relatively small. For larger scale systems with more DObjects nodes, the global schema can be replicated at a subset of the DObjects nodes such as landmark nodes.

Data operations and query languages. DObjects supports all standard data operations. Users can query, create, delete and update persistent entities. Both synchronous and asynchronous queries are supported. For synchronous queries, the system provides response immediately after its completion and execution of user's code is blocked during the query execution. For asynchronous queries, user can get results *incrementally* and operate on partial results while the query is being executed. In order to support the various data operations with transaction semantics, a variation of three-phase commit protocol is implemented.

The query language for our system could be implemented using any language that allows one to specify attributes or conditions for a given attribute in objects hierarchy. XPath or XQuery as well as OQL-like language are all valid approaches. DObjects also provides its internal *query language API* which strictly follows the object-oriented fashion of the data representation of persistent entities. A user creates queries by building a *hierarchy of objects*. Each query is created for a given persistent entity type and specifies which simple or referential attributes should be *populated*.

5.2 Privacy-Preserving Data Sharing

Having described all the components of the system for secure data sharing, we can now present an overview of the complete architecture. As discussed earlier, our building components include distributed anonymization protocol and distributed querying protocol that guarantees privacy of the individuals in the published data as well as data confidentiality of the data custodians, and the DObjects framework that provides a scalable and transparent query execution interface.

Architecture implementation proposal. The framework we are proposing is presented in Figure 6. The key point of the architecture is that DObjects nodes form a virtual database that can be used by clients. Despite the fact that physically the system is distributed, the virtual layer

provided by DObjects offers clients an abstraction of centralized system that can be used by submitting queries in the form of OQL. The goal is to provide a seamless access to data requiring anonymization as well as data not requiring anonymization. Therefore, when data not requiring anonymization is queried, the system simply returns result to the client. On the other hand, when data requiring anonymization is accessed, the framework transparently employs distributed anonymization and secure query execution that includes the secure union protocol to build responses. In order to support the access to anonymized data, DObjects nodes can form virtual groups (depicted as clouds in Figure 6). Note that for each object type requiring anonymization a separate virtual group is created.

When a client attempts to query the system, any DObjects node can be used as an entry point. It is the role of the framework to correctly deploy and execute query components and later return results. Therefore, the first DObjects node that receives OQL query performs an initial query decomposition. Parts of sub-queries that do not involve secure objects (i.e. objects that require anonimization) can be simply answered using usual database operators (joins, selections etc.). On the other hand, sub-queries that involve secure objects should be routed to virtual groups. It is the responsibility of each group to correctly execute such query for secure objects. Nodes within each group need to be capable of running distributed anonymization and querying protocols as described in Section 4 to guarantee data security and to prevent data ownership exposure. It can be observed that within each virtual group we employ secure data sharing scenario similar to the one presented in Figure 1c. The role of secure protocol coordinators is taken by each DObjects node within virtual groups. An interesting fact is that once the anonymized data is obtained from a virtual group, it can be later processed by any standard database operator. For instance, when anonymized patients need to be joined with another objects, a standard implementation of distributed join operator can be used.

It is also worth mentioning that the predicates specified in OQL for anonymized objects will have different semantics than those specified for the usual objects. In case of the anonymized data the predicates restrict results to objects that *possibly satisfy query predicates*. The reason for such behavior is that anonymization can remove exact values of attributes and in order to prevent data exposure, any predicate evaluation has to be performed after the data is anonymized.

Usage scenario. Let's now consider a sample scenario for deploying the system discussed above. We consider a group of cooperating hospitals. The patients databases maintained independently by each of the hospitals can be considered as a set of horizontally partitioned data distributed among multiple heterogeneous databases. Let's also consider a general statistics database about hospitals provided by other database system (such as data provided by the Online Analytical Statistical Information System project [4]) that can include data such as mortality/morbidity, population or general emergency room visits information. The DObjects configuration for such scenario can include County object, that has two referential attributes. The first referential attribute can be StatisticalData object that contains data from the

[4]http://oasis.state.ga.us/

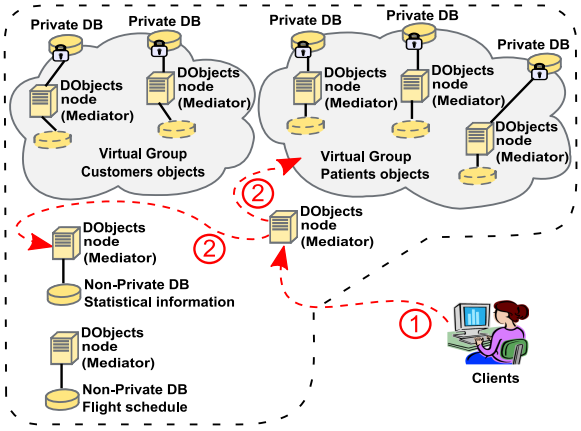

Figure 6: Implementation of architecture for secure data sharing.

```
select  data.mortality, data.morbidity, patients.age, patients.nation, patients.disease
from    County c, c.data data, c.patients patients
where data.mortality < 0.8 and patients.age > 60
```

Figure 7: Sample DObjects query.

publicly available source about the hospitals within given county. The second referential object can be a list of Patients that visit hospitals within each county. Under the HIPAA act, however, the personal health information is protected. Therefore Patient objects need to be anonymized before publishing. Instead of using the independent anonymization by each hospital, better results can be achieved by distributed anonymizaton protocol.

Let's now consider an OQL query submitted by user that is presented in Figure 7. A query execution process can be similar to one presented by red dashed arrows in Figure 6. First, the client submits query to any DObjects node. The system recognizes that three object types are queried: County, StatisticalData and Patients. The receiving node decides to produce County objects, however, the sub-queries for StatisticalData and Patients have to be submitted to other nodes. In case of the StatisticalData object, the data is directly provided by another DObjects node, and the corresponding sub-query is submitted to that location. In case of Patient objects, however, the data needs to be anonymized. In that case, the sub-query is submitted to a virtual group of nodes that will build a list of anonymized Patients data satisfying the query predicates.

6. SUMMARY

We have presented a set of protocols and a general architecture for sharing heterogeneous, distributed and possibly private data. The first protocol was a distributed anonymization protocol for privacy-preserving data publishing for horizontally partitioned databases based on the Mondrian partitioning method. The second protocol was a distributed querying protocol for securely computing the union of data tuples partitioned among nodes. The proposal of data sharing architecture was based on the principles of DObjects framework. In addition to providing a comprehensive solution for sharing secure data through employment of protocols mentioned above, the framework provides a gen-

eral query execution interface for scalable and secure access to heterogeneous and distributed data sources.

At present time we have implemented all the components of the basic DObjects framework. The implemented mechanisms within this framework, including dynamic and iterative query optimization and execution protocols and transaction management, have been experimentally evaluated. We have also performed some preliminary evaluations for the distributed anonymization protocol based on k-anonymity approach and secure union protocol used in secure query execution protocol. The results of experiments have proven that distributed anonymization protocol generates better anonymized data view when compared with independent anonymization of each database.

Our work continues along several directions. First, we are interested in developing a protocol toolkit incorporating more privacy principles and anonymization algorithms. Second, we are exploring different network topologies and optimization techniques in order to further improve the performance of the protocols. Third, we are interested in investigating game theoretic approaches to relax the semi-honest model assumption. Finally, we are planning to implement the virtual groups into DObjects data services to provide a working prototype of secure data sharing platform. We believe that such prototype will serve as good architecture for testing further ideas in this research area.

Acknowledgement

The work is partially supported by an Emory URC and an Emory ITSC grant. We would like to thank Kristen Lefevre for providing the implementation of Mondrian algorithm and the anonymous reviewers for their valuable comments.

7. REFERENCES

[1] C. C. Aggarwal. On k-anonymity and the curse of dimensionality. In *VLDB*, pages 901–909, 2005.

[2] R. J. Bayardo and R. Agrawal. Data privacy through optimal k-anonymization. In *ICDE '05: Proceedings of the 21st International Conference on Data Engineering (ICDE'05)*, pages 217–228, Washington, DC, USA, 2005. IEEE Computer Society.

[3] E. Bertino, B. Ooi, Y. Yang, and R. H. Deng. Privacy and ownership preserving of outsourced medical data. In *ICDE*, 2005.

[4] M. J. Carey, L. M. Haas, P. M. Schwarz, M. Arya, W. F. Cody, R. Fagin, M. Flickner, A. W. Luniewski, W. Niblack, D. Petkovic, J. Thomas, J. H. Williams, and E. L. Wimmers. Towards heterogeneous multimedia information systems: the Garlic approach. In *Proc. of the RIDE-DOM'95*, Washington, USA.

[5] B. C. M. Fung, K. Wang, and P. S. Yu. Top-down specialization for information and privacy preservation. In *Proc. of the 21st IEEE International Conference on Data Engineering (ICDE 2005)*, pages 205–216, Tokyo, Japan, April 2005.

[6] O. Goldreich. Secure multi-party computation, 2001. Working Draft, Version 1.3.

[7] S. Goldwasser. Multi-party computations: past and present. In *ACM Symposium on Principles of Distributed Computing*, 1997.

[8] R. Huebsch, B. N. Chun, J. M. Hellerstein, B. T. Loo, P. Maniatis, T. Roscoe, S. Shenker, I. Stoica, and A. R. Yumerefendi. The architecture of pier: an internet-scale query processor. In *CIDR*, 2005.

[9] V. S. Iyengar. Transforming data to satisfy privacy constraints. In *KDD*, pages 279–288, 2002.

[10] W. Jiang and C. Clifton. A secure distributed framework for achieving k-anonymity. *The VLDB Journal*, 15(4):316–333, 2006.

[11] P. Jurczyk and L. Xiong. Privacy-preserving data publishing for horizontally partitioned databases. Technical Report TR-2008-013, Emory University, Math&CS Dept., 2008.

[12] P. Jurczyk, L. Xiong, and V. Sunderam. DObjects: Enabling distributed data services for metacomputing platforms. In *Proc. of the ICCS*, 2008.

[13] D. Kifer and J. Gehrke. Injecting utility into anonymized datasets. In *SIGMOD Conference*, pages 217–228, 2006.

[14] D. Kossmann. The state of the art in distributed query processing. *ACM Comput. Surv.*, 2000.

[15] K. LeFevre, D. Dewitt, and R. Ramakrishnan. Incognito: Efficient full-domain k-anonymity. In *ACM SIGMOD International Conference on Management of Data*, 2005.

[16] K. LeFevre, D. DeWitt, and R. Ramakrishnan. Mondrian multidimensional k-anonymity. In *IEEE ICDE*, 2006.

[17] K. LeFevre, D. DeWitt, and R. Ramakrishnan. Workload-aware anonymization. In *SIGKDD*, 2006.

[18] A. Meyerson and R. Williams. On the complexity of optimal k-anonymity. In *PODS*, pages 223–228, 2004.

[19] L. Sweeney. k-anonymity: a model for protecting privacy. *International journal on uncertainty, fuzziness and knowledge-based systems*, 10(5), 2002.

[20] A. Tomasic, L. Raschid, and P. Valduriez. Scaling heterogeneous databases and the design of disco. In *Proc. of the ICDCS*, 1996.

[21] R. van Renesse, K. P. Birman, and W. Vogels. Astrolabe: A robust and scalable technology for distributed system monitoring, management, and data mining. *ACM Trans. Comput. Syst.*, 21(2), 2003.

[22] K. Wang and B. C. M. Fung. Anonymizing sequential releases. In *ACM SIGKDD*, 2006.

[23] K. Wang, P. S. Yu, and S. Chakraborty. Bottom-up generalization: a data mining solution to privacy protection. In *Proc. of the 4th IEEE International Conference on Data Mining (ICDM 2004)*, November 2004.

[24] X. Xiao and Y. Tao. Anatomy: Simple and effective privacy preservation. In *VLDB*, pages 139–150, 2006.

[25] Q. Zhang, N. Koudas, D. Srivastava, and T. Yu. Aggregate query answering on anonymized tables. In *ICDE*, pages 116–125, 2007.

[26] S. Zhong, Z. Yang, and R. N. Wright. Privacy-enhancing k-anonymization of customer data. In *PODS*, 2005.

[27] S. Zhong, Z. Yang, and R. N. Wright. Privacy-enhancing k-anonymization of customer data. In *PODS '05: Proceedings of the twenty-fourth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 139–147, New York, NY, USA, 2005. ACM Press.