A Survey of Privacy on Data Integration

Do Son Thanh

Abstract—This survey is an integrated view of other surveys on privacy preserving for data integration. First, we review the database context and challenges and research questions. Second, we formulate the privacy problems for schema matching and data matching. Next, we introduce the elements of privacy models. Then, we summarize the existing privacy techniques and the analysis (proofs) of privacy guarantees. Finally, we describe the privacy frameworks and their applications.

1 Introduction

1.1 Database Context and Challenges

Data integration: Data need to be collected from disparate data sites and integrated while sanitizing privacy-sensitive information.

There are some challenges:

- The data are protected by legitimate and widespread privacy concerns. Companies are afraid of being exploied by competitors or antitrust concerns.
- The data need to be integrated consistently while preserving privacy information.

Data integration has been a long standing challenge for the database community. This need has become critical in numerous contexts, including integrating data on the Web and at enterprises, building e- commerce market places, sharing data for scientific research, data exchange at government agencies, monitoring health crises, and improving homeland security.

Unfortunately, data integration and sharing are hampered by legitimate and widespread privacy concerns. Companies could exchange information to boost productivity, but are prevented by fear of being exploited by competitors or antitrust concerns.

Peer Data Management System (PDMS) This provides the extent to the classical data integration approach on large-scale network. The problem involves exchange data while passing through several intermediate peers. The mappings used in a PDMS have two main purposes: query translation and result translation.

There are some challenges:

- Preserving the privacy of the exchanged data while passing through several intermediate peers.
- Protecting the mappings used for data translation.
- Protecting the privacy without being unfair to any of the peers

Real-life systems include: Hyperion, PeerDB, BestPeer.

Federated Database System. Nowadays, there are a wide range of applications involving information access across databases (information sources) owned by different organizations. The most two common approaches to accomplish this are solutions offered in the area of *federated databases* and solutions employing *mediators*.

Federated database systems implement one-to-one connections between all pairs of databases that need to talk to each other. These connections allow one database system (requester) to query another database system (processor) in terms that the query can be interpreted correctly.

When mediator-based systems are used to support information access across heterogeneous databases, a mediator trusted by E1 and E2 generates and stores a mapping between the schemas of D1 and D2 to resolve the semantic heterogeneity. Essentially, mediators know data schemas.

There are some challenges:

- Federated database systems inherently require that D1 and D2 reveal their data schema (and the associated semantics), a main type of metadata, to each other. However, this requirement may raise serious privacy concerns when "there is an increasing need for sharing in- formation across autonomous organizations in such a way that no information apart from the answer to the query is revealed".
- Although a privacy control policy can be enforced by the mediators, such a solution has to rely on fully trusted and highly secure mediators to preserve the privacy and confidentiality of metadata. And such an approach is not (very) practical, since (a) building a highly secure mediator is not only very expensive but also very difficult, if not impossi- ble, because almost every host providing services could be hacked; (b) from the trust management point of view, such a continuous high trust requirement is very difficult to be satisfied, and as a result, such a mediator (third party) is unlikely to be deployed. (Fundamentally, the more trust you assume, the more vulnerable the system.) Nevertheless, the above discussion shows that preserving the privacy of metadata while enabling semantic interoperation is a diffi- cult problem, since, often, the technologies proposed for en- abling semantic interoperation depend heavily on insecure mediation based on the metadata.

Many applications require users from one organization to access data belonging to another organizations. Sharing

metadata, this may not be acceptable for certain organizations due to privacy concerns. For example, organizations like FBI and CIA may never want to reveal their metadata and divulge crucial information about what information is stored in their sources. Moreover, storing the schema on more systems obviously increases the threat to the confidentiality of the schema.

Data Mining: Works in privacy-preserving distributed data mining have studied how to train and apply classifiers across disparate datasets without revealing sensitive information at the datasets.

In this context, it is assumed that data integration (including record linkage) has already been done.

1.2 Provision Questions

There are many issues raised by database community:

1) *How can we develop a privacy framework for data integration that is flexible and clear to the end users?* This is first mentioned in [2] as a standard to build a privacy policy inside data integration system. Individual organizations may define their own policies to address their customers needs. The problems are exacerbated in a federated environment. The task of data integration itself poses risks, as revealing even the presence of data items at a site may violate privacy.
2) *All privacy views and policies must result in a single piece of privacy metadata?* In [2], the authors suggested that the privacy policies could be enforced by the server holding the data. As a subsequent, every data item leaving the server should be annotated with privacy metadata expressing the privacy policies that have to be applied. These annotations travel with the data, and are preserved and perhaps modified when the data is integrated with data from other sources or transformed.
   In the context of data integration among multiple data sources, query execution becomes much harder. It is not obvious how to retain a set of privacy views/policies in a single piece of privacy metadata (or in a single, multiple encrypted data instance).
3) *How can we develop schema matching solutions that do not expose the source data and schemas?* In [2], the authors pointed out that all current existing matching algorithms, however, assume that sources can freely share their data and schemas, and hence are unsuitable.
4) *How can we match entities and con- solidate information about them across sources, without revealing the origin of the sources or the real-world origin of the entities?*

2 Privacy Problems

2.1 Privacy-Preserving Schema Matching

In [2], to develop matching algorithms that preserve privacy, first the following components need to be developed:

2.1.1 Matchings Construction/Prediction

Statement. To create correspondences (instance matching or schema matching) without revealing data values, or even the schemas.

Solutions. Some solutions have been proposed:

- **Learning-based schema matching.** The idea is that one or more classifiers (e.g., decision tree, Naive Bayes, SVM, etc.) are trained at source S (by data instances or schema attributes of T), then sent over to target T for classification (on data instances or schema attributes of T). In some cases, we also perform in reverse direction (from T to S) to combine both classification results. Finally, the classification results are used to construct a similarity matrix between all attributes of S and T.
- **Pair-wise Mutual Information.** In [3], the authors proposed a protocol called "privacy-preserving schema matching using mutual information".
  Assumptions. Different schemas in the same domain have similar probability distribution of attributes and mutual information (MI) capturing the correlation between attributes.
  Method. Match MIs of two schemas using private set intersection and return intersected elements only. The mutual information between two attributes is a measure of the amount of information (e.g., entropy) that each attribute contains about the other attribute.
  Pros. Simple and direct with only two participants. Effective in many practical scenarios where schemas only differ in attributes' name and have highly similar structure.
  Cons. Two parties must join the protocol which relies on a reliable channel (in case of failure, might restart again).
- **Privacy preserving mediator.** When mediator-based systems are used to support information access across heterogeneous databases 3, a mediator trusted by E1 and E2 generates and stores a mapping be- tween the schemas of D1 and D2 to resolve the semantic heterogeneity. Essentially, mediators know data schemas.
  Implementations. PACT framework [12] is the pioneer in this approach.
  Pros. Flexible and Light-weight. The participants need not to know the implementation and it reduces the overhead cost incurred at each data source.
  Cons. Although a privacy control policy can be enforced by the mediators, such a solution has to rely on fully trusted and highly secure mediators to preserve the privacy and confidentiality of metadata. And such an approach is not (very) practical, since (a) building a highly secure mediator is not only very expensive but also very difficult, if not impossible, because almost every host providing services could be hacked; (b) from the trust management point of view, such a continuous high trust requirement is very difficult to be satisfied, and as a result, such a mediator (third party) is unlikely to be deployed.

2.1.2 Human Verification of Matches

Statement. The goal is to give (untrusted) users enough information to validate or repair low-quality correspondences, while preserving the privacy (data instances, schema structure, etc.) of involving data sources.

Solutions. There are some solutions:

- **Sample Data Values.** One way to achieve this can be randomly selecting some values for particular attributes and show the user only these values. It can be argued that revealing only few attribute values does not reveal anything useful about the characteristics (distribution, structure,...).
- **Relevant Attributes.** Another way is construct a small context by showing the user only relevant attributes/-correspondences.
  Pros. Users need not to be trusted.
  Cons. Firstly, a measure for privacy loss is needed in this context. Secondly, the verification should not be performed on all correspondences to avoid inferring attack by combination of multiple observations.

2.2 Privacy Preserving Data Matching

2.2.1 Record matching

This is a sub problem of data integration, mentioned in [17] (SIGMOD 2007).

Statements. There are two parties P and Q giving their relations R_P and R_Q to be matched. After the matching process, P only obtains the matching set P_{Match} related to R_P and Q only know the records of the matching set Q_{Match} related to R_Q.

Solutions. Secure Data Matching and Secure Schema Matching proposed in [17]. The authors designed a protocol using a third party to perform record matching. The matching result is obtain by mapping records to a vector space to preserve the privacy of data sources.

- **Pros.** Automated matching without revealing records or attributes. The quality of matching is also guaranteed with a good computational performance.
- **Cons.** Needs 3 parties (two main-players and one mediator) that must be semi-honest (participating in protocol but can arbitrarily perform information inference). Hard to implement due to distributed nature of protocol which requires a reliable channel and strongly coupling code.

2.2.2 Record linkage (Data Instance)

In data integration and sharing, the record linkage problem is mentioned in [2]. Record linkage can be viewed as a pattern classification problem. In pattern classification problems, the goal is to correctly assign patterns to one of a finite number of classes.

Statement. The goal of the record linkage problem is to determine the matching status (this is abstract definition) of a pair of records brought together for comparison.

We need solutions for the following sub problems:

- **Privacy-preserving record linkage:** that is discovering the records that represent the same real world entity from two integrated databases each of which is protected (en- crypted or anonymized). In other words, records are matched without having their identity revealed.
- **Record linkage aware data protection:** that is protecting the data, before sharing, using anonymization techniques that are aware of the possible use of record linkage, with public available data, to reveal the identity of the records.
- **Online record linkage:** linking records that arrive continuously in a stream. Real-time systems and sensor networks are two examples of applications that need online data analysis, cleaning, and mining.

2.3 Privacy Preserving Querying

2.3.1 Querying Across Sources

Statement. Once semantic correspondences have been established, it is possible to query (e.g., with SQL queries) across the sources. How do we ensure that query results do not violate privacy policy? How do we query the sources such that only the results are disclosed? How can we prevent the leaking of information from answering a set of queries?

Solutions. In [2], only a few general techniques exist today for querying datasets while preserving privacy:

- Statistical databases
- privacy-preserving join computation
- privacy-preserving top-K queries

3 Privacy Settings

3.1 Privacy Objects

There are various kinds of object to be protected in privacy preserving system/framework.

Data: cover all concrete data in database systems, including healthcare data, consumer data, personal data etc.

The service customer learns only the answer to the query, and not any of the data used to compute it.

Record: The information in data sources (database, file system, computer resources, etc.) that represents the real world entity. This is the main concern in data matching [17]. In some contexts, a record can be interpreted as data instance (a set of all tuples about an entity).

Schemas/Metadata. when performing schema matching, we need to protect the schema of two data sources. Protecting schema means that we do not reveal a certain degree (or number) of attributes of schema or completely hide information (name, structure, ...) of schema attributes.

Matchings. The mappings (between schema attributes or data values) used for data translation need to be protected to avoid revealing involving information and inferring attacks.

Queries. The service provider does not learn the query, only that a query was performed against a particular users information.

Anonymous communications. Service customers and service providers do not know who the opposite party is.

3.2 Privacy Concepts

Privacy is addressed today by preventing dissemination rather than integrating privacy constraints into the data sharing process. Privacy-preserving integration and sharing of research data in health sciences has become crucial to enabling scientific discovery.

HIPAA "Safe Harbor" De-Identification of Medical Record Information requires that each of the concerned identifiers of the individual or of relatives, employers, or household members of the individual must be removed from medical record information in order for the records to be considered de-identified.

Most privacy laws balance benefit vs. risk: access is allowed when there is adequate benefit resulting from access. An example is the European Community directive on data protection which allows processing of private data in situations where specific conditions are met.

Privacy Views. Each privacy view specifies a set of private attributes and an owner. By definition, data that appears in some privacy view is considered private; otherwise it is not private. A simple example of a privacy view is given below:

```
PRIVACY–VIEW patientAddressDob
OWNER Patient.pid
SELECT Patient.address, Patient.dob
FROM Patient
```

This privacy view specifies that a patients address and dob (date-of-birth) are considered private data when occurring together. Similar definitions are possible for fields that specify "individually identifiable information": Sets of attributes that can be used to tie a tuple or a set of tuples in a data source to a specific real-world entity (e.g., a person).

Privacy views could be implemented by a privacy monitor that checks every data item being retrieved from the database and detects if it contains items that have been defined as private. There are two approaches: compile-time (based on query containment) and run-time (based on materializing the privacy views and building indices on the private attributes). Both approaches need to be investigated and tradeoffs evaluated.

Privacy Policies (Access Control Policies). The database administrator can decide which policy applies to each view. More precisely, a policy defines the accessibility of private data for a specific or any beneficiary. Privacy policies can be enforced by the server holding the data: data items will be shared only if the purpose statement of the requester (see below) satisfies the policy. But, in addition, every data item leaving the server should be annotated with privacy metadata expressing the privacy policies that have to be applied. These annotations travel with the data, and are preserved and perhaps modified when the data is integrated with data from other sources or transformed.

Example:

```
PRIVACY–POLICY individualData
ALLOW–ACCESS–TO y
FROM Consent x, patientAddressDob y
WHERE x.pid = y.owner and x.type = 'yes'
BENEFICIARY y.owner
```

Purpose Statements. A flexible language is required in which applications can state the purpose of their action, and explicitly mention the beneficiary.

3.3 Datasets

Healthcare. In [2], the authors use healthcare scientific data as a leading example.

FBI and CIA. used in [12]. In FBI and CIA databases, each table contains 50 to 10000 records depending on its functionality, and the corresponding access control rules are stored in a separate authorization table. The authors designed the organizational ontology for each organization. Each ontology has about 20 classes and about 300 triples if they are represented in N-TRIPLE format. Each organization has 5 roles. Each role has different privileges to access the tables in database.

PET. In [12], we use a dataset obtained from a pet hospital chain to conduct our experiments. The original data set includes information about the demographics of pets, their hospital visits, diagnosed diseases, lab tests, prescribed medications, etc. The dataset contains information about over 45 million pets with hospitals spanning 40 US states. For the sake of the experiments, we modified the data set as follows. First, we partitioned the data based on the state information to obtain 40 different dataset partitions. Second, for each partition, we mapped the data into a smaller schema, similar to the ones shown in Figure 1, thus keeping only the pet medication information. We assumed that each partition represents the database of a peer in the network. In particular, we used six such partitions representing six different states. Table 2 shows some statistics about these dataset partitions.

CENSUS. Used in [19]. (downloadable from http://www.ipums.org), containing personal information of 50K individuals with 8 dimensions (average domain size 28.1).

ADULT. Used in [19]. containing personal information of 32,561 individuals with 15 dimensions (average domain size 1476.4).

Three datasets. Used in [17], consists of British Columbia voters, personal data by a public administration, and business data owned by a public administration.

4 Privacy Techniques

This sections describes core preserving privacy techniques that are used to protect privacy objects (as described in 3.1). Table 1 provides the summary of privacy techniques mentioned on PPP, PD protocol and PACT framework.

4.1 K-Protection

K-Protection is the technique used for protecting query results from unauthorized parties in PDMS setting. It allows the client (user) to determine the privacy level (k value) that

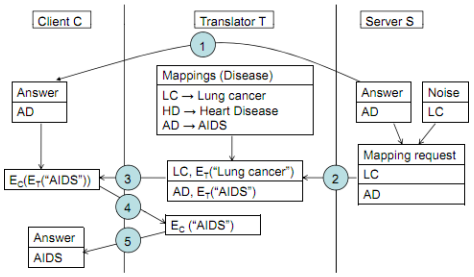

Fig. 1: Execution of PPP with k = 2. The messages between peers are: (1) S → C: query result in the context of S. (2) S → T: mapping request. (3) T → C: encrypted mappings. (4) C → T: double-encrypted mapping. (5) encrypted mapping that can be decrypted by C

means the untrusted parties cannot have certainty of each value of the query result greater than 1/k.

The definition of k-protection is described in [4] as follows:

Definition 1: **(k-protection)** *Let R^X, D^X, UR^X be the query answer, domain of the query answer, and the set of unique values in the query answer respectively in the context of X. Let $R(\pi; X)$ be the information observed by the peer X during the execution of the protocol π. A protocol π is said to provide k-protection if for each T_i, $i \in [1,t] Pr[v \in R^{T_i} | R(\pi, T_i)] \leq \frac{1}{k}$ for all $v \in D^{T_i}$.*

Example 1: Figure 1 shows a simple example of PPP Protocol [4] with one client C, one server S, one translator T and k-protection = 2. Assume the answer of query from client C has only one value AD (in the server's schema). Before passing the query result to the translator T, the server S inserts an additional fake value to the result LC to make a mapping request (Step 2). Therefore, the translator cannot determine whether AD or LC is the real result with the probability exceeds 1/2.

4.2 Schema Obfuscation

While K-Protection is designed to protect the query result at server-side, Schema Obfuscation on the other hand is used to preserve the privacy of the client's schema. When a query is issued by the user, the query is transformed based on the user ontology instead of the client's schema. Therefore the client's schema will be kept private.

Example 2: Figure 2 illustrates two ontology of FBI and CIA database. Assume we have a meta-data that indicates the ontology term $staff$ corresponds with the database table-name $adminpersonnel$. To obtain information from FBI database, an user at CIA issues a query:

select $fname, lname, compensation$ **from** $adminpersonnel$ **where** $compensation > 70000$;

This query is automatically "obfuscated" by using CIA's ontology. PACT replace table-name $adminpersonnel$ by ontology class $staff$, and the schema terms $fname, lname, compensation$ by ontology terms $fistName, lastName, pay$, respectively. The re-written query now is:

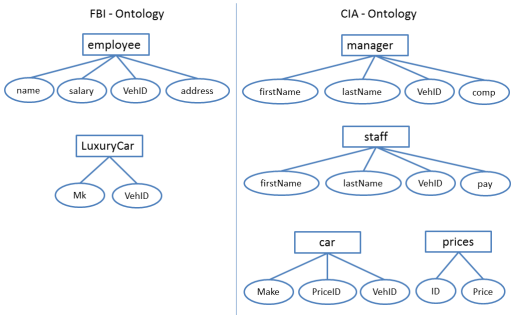

Fig. 2

select $fistName, lastName, pay$ **from** $staff$ **where** $pay > 70000$;

4.3 Data Encryption

One straight forward method is used to protect privacy is encryption. Encryption is commonly collaborated with other techniques to enhance privacy. The reason is that the techniques such as k-protection, schema obfuscation are designed to protect a particular kind of data. Consequently, the remaining vulnerable information could easily be secured by using encryption. Therefore, the object of encryption can be all privacy objects (as described in **??**) such as mappings, query or result. Furthermore, since the purpose is to aid other techniques the implementation of encryption depends on the design of the protocol or framework. In what follows we will describe some encryption implementations in some protocol and framework.

4.3.1 PPP Protocol [4]

PPP protocol only encrypt translator mappings to fulfil *fairness* requirement (i.e.; only result-related mappings can be showed to the client). The institution behind this requirement is to access the translator's mapping may cost money. Consequently, it is may be unfair if the client must pay for useless mappings. Besides of that, the translator must not know what mapping the client selected to request. To solve these problems, PPP uses *commutative encryption* technique to encrypt mapping.

Figure 1 shows an example of PPP protocol where translator's mappings are encrypted (only the answer of mapping) then send to the client C along with the query result received from the server S (Step 3), here $LC \rightarrow E_T("Lung\_Cancer")$ and $AD \rightarrow E_T("AIDS")$. The commutative encryption ($E_C(E_T(M)) = E_T(E_C(M))$) allows the client get decrypted answer by sending the double-encrypted request $E_C(E_T("Lung\_Cancer"))$ to the translator then get back the result $E_C("Lung\_Cancer")$ after the translator applies decryption.

4.3.2 PD protocol [19]

PD protocol works on the same setting as PPP protocol but uses *oblivious transfer* to replace for *commutative encryption* due to its security problems. Oblivious transfer

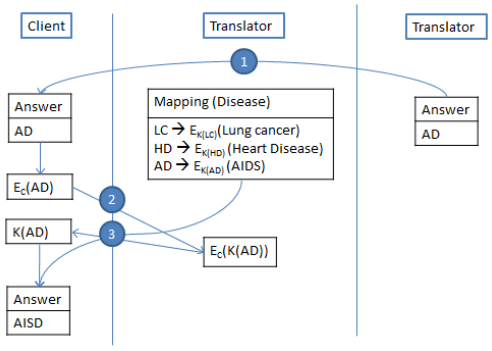

Fig. 3

(OT) [] is a probabilistic approach to allow the client retrieve mappings without letting the translator know what mapping the client chosen.

A simple example of PD protocol is shown in Figure 3. The right part each mapping entry is encrypted by a special encryption function that key is depend on the left part. To get the final result, the client encrypt result received from the server AD (Step 1) then send mapping request $E_C(AD)$ to the translator (Step 2). The special designed of the encryption algorithm $E_T(E_C(x)) = E_C(K(x))$ allow the translator send back the key of selected entry mapping $E_C(K(AD))$ without knowing the content of request together with translator mapping table (Step 3). From these information, the client can obtain the correct mapping $AD \rightarrow AIDS$ from the mapping table.

4.3.3 PACT [12]

Whereas the translator in PPP and PD protocol is trusted (i.e.; the client and server share their schemas with translators), the mediator in PACT setting is untrusted. Consequently, the schemas and mappings are sensitive information and need to be protected. To resolve this problem, both mappings and client query are encrypted.

Mapping encryption: Mappings are created by experts and encrypted in off-line processing phase. Note that these mappings are ontology mappings.

Example 3: Here are some simple examples based on ontology of FBI and CIA database as illustrated in Figure 2:

- ($E_{K_{FBI}}$ (employee), $E_{K_{CIA}}$ (manager), subClassOf)
- ($E_{K_{FBI}}$ (employee), $E_{K_{CIA}}$ (staff), subClassOf)
- ($E_{K_{FBI}}$ (employee).$E_{K_{FBI}}$ (salary), $E_{K_{CIA}}$ (manager).$E_{K_{CIA}}$ (comp))
- ($E_{K_{FBI}}$ (employee).$E_{K_{FBI}}$ (salary), $E_{K_{CIA}}$ (staff).$E_{K_{CIA}}$ (pay))
- ($E_{K_{FBI}}$ (Name), Merge($E_{K_{CIA}}$ (fisrtName), $E_{K_{CIA}}$ (lastName))

Query encryption: In parallel, a query will be encrypted using two keys: (a) the client master key; and (b) a predetermined session key (denoted K_S) between the client and the server.

Example 4: Continue with CIA and FBI databases setting, to obtain information from CIA database, an user at FBI poses a query:

select $name, salary$ **from** $employee$ **where** $salary >$ 70000;

All ontology terms (e.g., employee and salary) in the query is encrypted by using FBIs master key, denoted K_{FBI}, and all the values (i.e., 70000) in the query is encrypted by using the session key K_S.

The encrypted query:

select $E_{K_{FBI}}(name)$, $E_{K_{FBI}}(salary)$ **from** $E_{K_{FBI}}(employee)$ **where** $E_{K_{FBI}}(salary) > E_{K_S}(70000)$.

The final query after applying mapping tables as shown in Example 3:

select Merge($E_{K_{CIA}}(firstName), E_{K_{CIA}}(lastName)$), $E_{K_{CIA}}(comp)$ **from** $E_{K_{CIA}}(manager)$ **where** $E_{K_{CIA}}(comp) > E_{K_S}(70000)$;

union

select Merge($E_{K_{CIA}}(firstName), E_{K_{CIA}}(lastName)$), $E_{K_{CIA}}(pay)$ **from** $E_{K_{CIA}}(staff)$ **where** $E_{K_{CIA}}(pay) > E_{K_S}(70000)$;

Technique	PPP	PD	PACT
K-protection	√	√	
Schema Obfuscation			√
Data Encryption	√	√	√

TABLE 1: Privacy techniques

5 Privacy Analysis

5.1 Parameters

The experiments often show performance results in the impact of following parameters:

- Result data size
- The number of users
- The number of access control rules

5.2 Metrics

Privacy Disclosure. In [2], the authors suggested to consider reliable metrics for quantifying privacy loss. In the original definition $H(y)$ corresponds to entropy of y, and $H_x(y)$ corresponds to conditional entropy of y given x then privacy loss due to revelation of x is given as follows:

$$Infer(x \rightarrow y) = \frac{H(y) - H_x(y)}{H(y)} \quad (1)$$

Security Elements. analyzed in [17] in terms of disclosure of participated parties, including:

- length of database records
- database size
- set of matching records
- set of matching attributes
- number of matching attributes

Processing Performance. mostly concern about query performance of the database system in the presence of privacy-preserving techniques. Somes to be mentioned:

- End-To-End Response Time
- Component Throughput
- Matching Time: the computational time to obtain matching result.

Query Selectivity.

Matching Quality. reflects the quality of schema or data matching results when privacy techniques are applied. In most cases, they simply use precision and recall. A good technique is expected to produce high quality matching while maintaining a high level of privacy.

5.3 Types of Attack

In [4], the author discuss seven type of attacks against the protocol in e case of a malicious model, where peers may deviate from the protocol to launch attacks against other peers. The first four attacks are related to the privacy of the query result, while the following three attacks related to the privacy of the mapping tables.

5.3.1 Query Replay Attack

In a query replay attack, if the same query was issued several times by the same or different clients to the same server S, and passing through the same translator T, then T may attempt to learn more information about the result, each time the query is issued.

Solution: insert same noise for the same query result to prevent attackers from inferring the intersection of two different query results.

5.3.2 Known Result Attack

In this attack, a translator T may send queries to S whose results are already known to T, in an attempt to learn how S selects noise values.

Solution: One approach might be increasing the k-value when an attribute is queried several times (greater than pre-defined threshold). Thus, the probability of retrieving complete noise data is very low.

5.3.3 Privacy Relaxation Attack

A translator T_h, $h \in [1, n]$ may attempt to reduce the k-value of k-protection before forwarding the query to T_{h+1}, Hence, T_h can illegitimately relax the privacy requirement set by C.

Solution: the client C sends raw query (based on client schema) and k-value (of k-protection) directly to the server S and not to any translator. Moreover, every translator T_h sends to S the attribute name mappings it used for translating the query. However, establishing the direct connection might incurs huge workload and resource consuming.

5.3.4 Translators Collusion Attack

In this attack, several translators may collude together in an attempt to gain additional knowledge about the result. For example, in PPP protocol some translator can collect all their received mapping requests that contain query result and noise in a central location for extracting real result from noise.

Solution: Use encryption or ensure k-protection for all data passing through translator.

5.3.5 Mappings Correlation Attack

In this attack, the client may want to infer other mappings by correlating two consecutive response message from two translators T_{h+1} and T_h. For example, in PD protocol, with two sequential response message such as $(x_1 \rightarrow E_{T_h}(y_1), x_2 \rightarrow E_{T_h}(y_2)$ and $(y_1 \rightarrow E_{T_{h+1}}(z_1), y_2 \rightarrow E_{T_{h+1}}(z_2))$. The client can easily infer the mappings $(x_1 \rightarrow y_1, x_2 \rightarrow y_2)$ because of the order of message is preserved through translators.

Solution:

- Randomly shuffled mapping request message (contained query result & required mappings for translating) before passing to next translator.
- Encrypt translators' mappings then break key into multiple pieces, each piece send to each translator.

5.3.6 Rick Client Attack

In this attack, a rich client C may decide to purchase all the mapping entries from a certain translator T. This way, it can clone T and provide the same translation services, which T previously offered. Thus, T can be deprived from potential future revenues.

Solution: Add rule to term of services or customer agreement when providing mappings.

5.3.7 Black Market Attack

This attack refers to the situation where clients choose to form a black market to exchange mapping entries and thus negatively impact the revenues of the translators.

Solution: No solution yet.

6 Privacy Frameworks

6.1 General Privacy Middlewares (both client-server and P2P context)

Care Middleware by Pareschi. Implemented some techniques:

- Anonimity
- Obfuscation
- Policy
- Cryptography

AnonySense by Kapadia. Implemented some techniques:

- Anonymity
- Obfuscation
- Cryptography

6.2 Peer Data Management System (PDBMS)

There are a wide range of real-life systems:

PeerDB. PeerDB is a P2P distributed data sharing system, which supports fine grained content-based searching and does not rely on a shared schema. It employs information retrieval approaches, such as keyword search, to find peers having data relevant to the users query. It also uses mobile agents to be able to perform operations at peer sites.

coDB. coDB is another P2P database system with techniques for searching and updating peer databases. In coDB, mappings are defined in terms of GLAV coordination rules.

Piazza. Introduced by Halevy et al, Piazza PDMS addresses the problems of defining a language for schema mappings, reformulating queries based on these mappings and automating the schema matching process in order to assist the user in constructing the mappings.

Piazza is a Peer Data Management System (PDMS). Unlike Hyperion, it does not support value-to-value mappings.. It uses more complex schema mappings. In particular, Piazzas mapping language supports both Local-As-View (LAV) and Global-As-View (GAV) mapping types.

Hyperion. Hyperion is a Peer Data Management System (PDMS). The mappings used in a PDMS have two main purposes: query translation and result translation. For query translation, different types of mappings were proposed.

The Hyperion system is based on the notion of mapping tables, which encompass attribute-to-attribute and value-to-value mappings.

HePToX. the HePToX PDMS was introduced primarily for XML sources. In HePToX, mapping rules between peers can be inferred given simple correspondences between their schema elements. The rules are expressed using a language called TreeLog, which can handle data-metadata mappings. This whole body of work did not consider the security issues in PDMSs.

6.3 Privacy Preserving Protocol

In [4], the authors presented a novel query answering protocol deployed on top of the Hyperion system. The proposed protocol has two specific goals: (1) hiding the identity of the real values in the query result as they pass through the intermediate translator peers, and (2) ensuring that no mapping entries other than the ones needed to translate the query result get disclosed to the client. Recall that if the second goal is not met, then the protocol will be unfair.

The PPP operates in three phases:

Phase I. Query Delivery
Phase II. Result and Mappings Collection
Phase III. Mappings Decryption and Result Translation

6.4 Light-weight privacy preserving protocol

In [19], we stick to the strict requirement that no party can observe the mappings of other peers and develop two lightweight protocols: a simple method based on serial translation and a more complex one that supports parallel translation. Furthermore, we consider a stronger adversary model where there may be collusions among peers and propose an efficient protocol that guards against collusions. We conduct an experimental study on the performance of the proposed protocols using both real and synthetic data. The results show that the proposed protocols not only achieve a better privacy guarantee than PPP, but they are also significantly more efficient, because they do not rely on expensive cryptographic operations.

6.5 PACT

Privacy-preserving Access Control Toolkit (PACT), proposed by [12], is a novel solution that enables privacy-preserving secure semantic access control and allows sharing of data among heterogeneous databases without having to share metadata. PACT uses encrypted ontologies, encrypted ontology-mapping tables and conversion functions, encrypted role hierarchies and encrypted queries. The encrypted results of queries are sent directly from the responding system to the requesting system, bypassing the mediator to further improve the security of the system. PACT provides semantic access control using ontologies and semantically expanded authorization tables at the mediator. One of the distinguishing features of the PACT is that it requires very little changes to underlying databases. Despite using encrypted queries and encrypted mediation, PACT provides acceptable performance.

PACT focuses on metadata instead of data; and PACT focuses on information access instead of information integration.

PACT has two phases: (i) the offline phase - the initial processing that takes place before any query is processed; and (ii) the online phase, which shows how an inter-organization query is processed in runtime.

The offline procedure of PACT is to (1) translate the (syntactic) access control policy of each organization to a semantic access control policy against the organizations ontology, and (2) prepare the other metadata used by the mediator.

The online procedure of PACT consists of following steps:

Step 1. Schema Obfuscation and Query Encryption
Step 2. SQL Query Parsing.
Step 3. Encrypted Query Rewriting
Step 4. Semantic Access Control
Step 5. Semantic to Syntactic Query Translation
Step 6. Query Evaluation
Step 7. Returning the Results

6.6 Privacy Preserving Ontology Matching Framework

This framework originates from 2005 in the AAAI workshop on Context and Ontologies. In [11], the authors proposed a framework for interoperation systems, which leverage two privacy-preserving ontology matching algorithms:

Automated. This algorithm enables interoperation (querying) between two organizations A and B. The mapping rules used for interoperation are created by automated ontology matchers and stored at the mediator in a mediated system.

- *Assumptions:* Use mediator to store mapping rules. The mediator is *untrusted* (it cannot know the key to decrypt the queries and mapping rules). The ontology matcher operates on encrypted ontologies only. Two organizations should trust each other.

- *Methodology:* Queries are encrypted by a symmetric private key used only by A and B (K_{A-B}). Mapping rules are also encrypted by K_{A-B}. Ontologies are also encrypted. The encrypted mapping rules are created by using work similarity matching technique on a encrypted dictionary.
- *Pros:* Fast and simple. Word-based matcher and symmetric encryption are trivially implemented.
- *Cons:* Firstly, the matching quality might be poor due to the lack of human verification (human experts cannot decipher encrypted ontologies). Secondly, two organizations use the same private key. Thus, one organization can interfere the communication between the mediator and the other organization to obtain the other's ontology.

Semi-automated. This algorithm extends the automated algorithm with human-involving in matching process and asymmetric encryption for two organizations.

- *Assumptions:* Use asymmetric encryption system: each organization own a unique private key and publicize the public key to certified authority. Human expert has access to clear-text ontologies for semi-automated ontology matching.
- *Methodology:* Firstly, each organization sends its ontology encrypted by session key (K_s) to the expert. The expert use K_s to decrypt two ontologies and perform matching. Then the mapping rules are encrypted by public key: the terms of ontology A and B are encrypted by public keys K_A and K_B respectively. Finally, the encrypted mapping rules are stored at the mediator. When interoperating via the mediator, the queries and ontologies of each organization are encrypted by its public key. The mediator will rewritten the queries based on encrypted mapping rules.
- *Pros:* Two organizations do not need to trust each other. Semi-automated process can leverage other matching techniques (structural, pattern, inference, etc.) since the clear-text ontologies are accessible by the expert.
- *Cons:* The human expert must be trusted.

6.7 Secure Set Intersection Protocols for Data and Schema Matching

The first protocol is introduced in [17] (SIGMOD 2007) to perform secure data matching and schema matching. The principle is that by embedding the records into a vector space, any information about the records is hidden from the matching process. This idea is similar to learning-based schema matching which relies on a classifier to build the similarity matrix.

Assumptions. Three parties (two players and one mediator) are semi-honest (following protocol but might inferring information). Two main players have different privacy requirements also at schema level.

Methodology. The protocol consists of three main phases: (1) setting of the embedding space, (2) embedding of relations R_P and R_Q values and (3) comparison to decide matching records. For the sake of simplicity, we list some main privacy techniques:

- At some steps, two main parties use symmetric key encryption to prevent the mediator from seeing the plain name of schema attributes.
- Use threshold, set intersection and vector concept to embed the similarity of matching records into normalized distance between vectors. Thus does not reveal any information about data records and schema attributes.

The second protocol proposed in [3] does not require the third party (mediator) in the protocol. The authors proved that their privacy-preserving schema matching protocol is secure against malicious adversaries for three mapping types: one-to-one, onto and partial. One of the building blocks of their protocol is the privacy-preserving set intersection scheme. In the case where all the attribute entropies in one of the schemas are different from one another, the protocol executes a linear number of privacy-preserving set intersections.

6.8 Service Broker Architecture

In [1], the authors proposed a privacy preserving framework that leverages a service broker to process queries without revealing any useful information to the data sources or to the third parties. The broker is assumed to be secure and fully trusted to safeguard the privacy of organisational data. The service broker is responsible for both the query execution as well as the decision of determining which queries should be permitted for execution. The broker ensures that results of the query do not contain any information that the asker is not authorized to discover. The broker uses an algorithm called semantic request mediation based on global ontology to translate customer queries so that the semantic heterogeneity between organisation A's schema and N's schema can be resolved.

6.9 Secure Multi Party Computation (SMC)

SMC is a generative tool in cryptography. It is employed by several works on privacy issues in data mining. Oblivious transfer is also a SMC protocol (specifically, secure two-party computation).

Generally speaking, each party i in SMC has its own input x_i and the parties want to cooperate to calculate $f(x_1, ..., x_n)$ without any party i learning anything beyond $f(x_1, ..., x_n)$ and its own input x_i.

6.10 Private Information Retrieval (PIR)

PIR does not protect the mappings of the translators. For example, PIR allows the client to see the whole mapping table in plaintext, which is undesired in our problem. In oblivious transfer, the client learns M_σ only.

7 Related Work

Some of the privacy issues have been addressed for the case of a single database manage- ment system in Hippocratic Databases. Other privacy issues have been addressed for the case of a single interaction between a user and a Website in the P3P standard. None of the current techniques address privacy concerns when data is exchanged between multiple organizations, and transformed and integrated with other data sources.

8 Conclusions and Future Work

This survey is an integrated view of other surveys on privacy preserving for data integration. Interested readers can follow the references pin-pointed in this survey for further investigation. While the survey is intended for data integration [5], [13], [6], [14], many privacy models and techniques can be applied in other fields, such as information retrieval [18], [16], data mining, sensor network [8], [7], and crowdsourcing [10], [15], [9].

References

[1] F. Al-Neyadi and J. Abawajy. A privacy preserving service broker architecture for data sharing. *Future generation information technology*, pages 450–458, 2010.

[2] C. Clifton, M. Kantarcioglu, A. Doan, G. Schadow, J. Vaidya, A. Elmagarmid, and D. Suciu. Privacy-preserving data integration and sharing. *DMKD*, page 19, 2004.

[3] I. F. Cruz, R. Tamassia, and D. Yao. Privacy-preserving schema matching using mutual information. In *Proceedings of the 21st annual IFIP WG 11.3 working conference on Data and applications security*, pages 93–94, Berlin, Heidelberg, 2007. Springer-Verlag.

[4] H. Elmeleegy, M. Ouzzani, A. Elmagarmid, and A. Abusalah. Preserving privacy and fairness in peer-to-peer data integration. *SIGMOD*, page 759, 2010.

[5] A. Gal, M. Katz, T. Sagi, M. Weidlich, K. Aberer, H. Q. V. Nguyen, Z. Miklós, E. Levy, and V. Shafran. Completeness and ambiguity of schema cover. In *CoopIS*, pages 241–258, 2013.

[6] A. Gal, T. Sagi, M. Weidlich, E. Levy, V. Shafran, Z. Miklós, and N. Q. V. Hung. Making sense of top-k matchings: A unified match graph for schema matching. page 6, 2012.

[7] N. Q. V. Hung, H. Jeung, and K. Aberer. An evaluation of model-based approaches to sensor data compression. *TKDE*, pages 2434–2447, 2013.

[8] N. Q. V. Hung, S. Sathe, D. C. Thang, and K. Aberer. Towards enabling probabilistic databases for participatory sensing. In *CollaborateCom*, pages 114–123, 2014.

[9] N. Q. V. Hung, N. T. Tam, Z. Miklos, and K. Aberer. On leveraging crowdsourcing techniques for schema matching networks. In *DASFAA*, pages 139–154, 2013.

[10] N. Q. V. Hung, D. C. Thang, M. Weidlich, and K. Aberer. Minimizing efforts in validating crowd answers. In *SIGMOD*, pages 999–1014, 2015.

[11] P. Mitra, P. Liu, and C. Pan. Privacy-preserving ontology matching. *AAAI Workshop on Context and Ontologies*, 2005.

[12] P. Mitra, C.-C. Pan, P. Liu, and V. Atluri. Privacy-preserving semantic interoperation and access control of heterogeneous databases. *ASIACCS*, page 66, 2006.

[13] H. Q. V. Nguyen, T. K. Wijaya, Z. Miklós, K. Aberer, E. Levy, V. Shafran, A. Gal, and M. Weidlich. Minimizing human effort in reconciling match networks. In *ER*, pages 212–226, 2013.

[14] Q. V. H. Nguyen, X. Luong, Z. Miklos, T. Quan, and K. Aberer. Collaborative schema matching reconciliation. In *CoopIS*, pages 222–240, 2013.

[15] Q. V. H. Nguyen, T. Nguyen Thanh, T. Lam Ngoc, and K. Aberer. An evaluation of aggregation techniques in crowdsourcing. In *WISE*, pages 1–15, 2013.

[16] T. T. Nguyen, Q. V. H. Nguyen, M. Weidlich, and K. Aberer. Result selection and summarization for web table search. In *ICDE*, pages 231–242, 2015.

[17] M. Scannapieco, I. Figotin, E. Bertino, and A. K. Elmagarmid. Privacy preserving schema and data matching. In *Proceedings of the 2007 ACM SIGMOD international conference on Management of data*, SIGMOD '07, pages 653–664, New York, NY, USA, 2007. ACM.

[18] N. T. Tam, D. C. Thang, N. Q. V. Hung, and K. Aberer. An evaluation of diversification techniques. In *DASFAA*, pages 215–231, 2015.

[19] Y. Zhang, W. Wong, and S. Yiu. Lightweight Privacy-Preserving Peer-to-Peer Data Integration. *No Info*, pages 1–22, 2011.