Scientific Journal Impact Factor (SJIF) : 3.518
International Journal of Modern Trends in Engineering and Research
ISSN (ONLINE) : 2349 - 9745 ISSN (PRINT) : 2393 - 8161

Privacy Preserving Collaborative Query Service through Horizontal Data Integration

Subrata Bose

Department of Computer Science and Engineering
Neotia Institute of Technology, Management and Science, Kolkata, India

Abstract — Query service for information on demand is a natural requirement. This paper proposes a privacy preserving collaborative query service. A service provider provides this service to its customer by integrating data from multiple heterogeneous and autonomous data sources each of which holds a horizontal subset of the query specific data. The customer and the data sources are unknown to each other. The service provider acts as a bridge between them for servicing the query while preserving privacy. Secure data exchange by commutative encryption is the basic cryptographic approach considered in this work.

Keywords — collaborative computing, commutative encryption, security, secure message transfer, graded query privacies

I. INTRODUCTION

Collaboration is a natural phenomenon for solving complex problem which demands participation of multiple organizations. These organizations work together towards a common interest. Many information-intensive tasks from health and legal area benefit from collaboration [24]. Collaboration is useful and unavoidable in many situations, information retrieval being one of them. Retrieving data from multiple sources for information retrieval [18] is one such collaborative task which requires collaborative efforts of willing and capable organizations [19]. For example, for making a comprehensive tour plan for a family at Bangkok during festival season requires information sharing by multiple sources such as Airlines, Hotels, and Tour Operators etc. The range of collaborative applications is wide and growing. Today's world is information-intensive and most organizations are ICT enabled and information rich. Information retrieval service by a service provider by assimilating data from multiple sources is envisaged. In this paper we propose a platform for organizations for such a collaborative service which mutually benefits all of them and at the same time safeguard their privacy.

Service providers provide services on demand over the Internet. Typically, these services use large databases and powerful servers that host web applications and services. Anyone with a suitable internet connection and a browser can access these applications [4]. Getting information at a mouse click in a browser is helpful for any individual vis-à-vis the time and effort for searching, collecting and collating information from different data sources subject to availability. We are considering one such web service – a *collaborative query service*. A set of data sources allows a service provider to offer this service to its customers on their databases. Customer issues query to the service provider who in turn finds potential and willing data sources relevant to solve the query with the help of their data. The data sources should generally have prior arrangement with the service provider though some of them could be spotted on the fly and included in the query service depending on the need [18, 19]. The databases maintained independently by different organizations participating in this query service are not expected to be homogeneous. Database software products, their versions and schema are likely to vary from one data source to the other.

Query service can be provided in different domains like banking and finance, insurance, airlines, science, art and culture, history, literature, sports, cinema, crime records, company related information etc. (Table 1). Service provider decides on the list of information domains depending on

its business interest and potential.

Table 1. Examples of queries and related privacy concerns

Domain	Query	Possible privacy concerns
Airlines	Find list of young men who had flown from City-A to City-B on a particular day by morning flight	Gender and Age limits of the passengers, Name of the cities, Date and Time of the flights
Banks	List all bank account details of person X having minimum balance of M million in the accounts as on a particular date	Person's identity in any form such as Name, SSN No, Minimum balance amount, Bank's identity such Name, Address, Country etc.
Crime records	Find the crime records of a person for crime committed between Year-1 and Year-2 from Police and CID departments' records of a state	Person's identity in any form such as Nationality, Name, Identification mark, Physical measure such as Height, Weight, Values of years, Name of the State, Name of Departments

The customer who submits queries to the service provider should not care to know who the data sources are and the data sources should not bother about the identity of the customer and other data sources. However, all the players have to reveal their information in such an open environment. Needless to say *privacy* is a serious concern for all concerned – customer for its query and identity, the data sources for their data, result and identities. The service provider acts here as an intermediary. It receives query from the customer and answers it with the help of data sources without gaining any knowledge about the query, data and the result. Information retrieval as a Service – a cloud based IR service has been proposed in the literature [18, 19]. The service provider acts as an integrator of multi source data. In general, the integration may be horizontal or vertical or mixed (any combination of these two) like in distributed database. The current work offers *privacy solution* for collaborative query service by integrating data from multiple sources where each data source holds a *horizontal subset* of query specific data. Commutative encryption is the basic cryptographic technique adopted in this work.

Commutative encryption is an important tool used in many cryptographic protocols [8, 30]. In commutative encryption if a plaintext message m is encrypted by number of different keys in any order, they are mapped onto the same cipher text. Thus if some data is encrypted more than once, the order in which it is decrypted does not matter. In other words, the order of encryption need not be followed for decryption (Section 4.1). We have used this basic property of commutative encryption for secure data exchange between any two parties via a third party and this forms the basis of horizontal data integration for the proposed query service.

II. QUERY SERVICE AND ITS PRIVACY

2.1 Customer Query Privacy

The customer issues a parametric query to the service provider through a web application program interface. The application program accepts customer's choices and data through the interface and generates a dynamic SQL query [11]. The dynamic parametric query has placeholders denoted by question marks (?) in the query text (Fig 1). The textual content of such query normally does not reveal any privacy but the complete query with the values of the parameters replacing the placeholders makes it sensitive [9]. This is a concern for the customer. The problem of preserving query privacy is equivalent to that of hiding the parameter values (query constants) from the service provider and optionally from the databases. Hiding the constants during query processing requires cryptographic technique which increases computation and communication overheads. As a counter measure the system offers flexibility of attaching different levels of protection to the query constants rather than treating all of them at per. This helps reduce cryptographic overheads in practical situation.

```
SELECT airline,flight_number,departure_time,journey_date FROM Airlines
  WHERE City-A = ? AND City-B = ? AND
        departure_time <= ? AND journey_date BETWEEN ? AND ?
```

Fig 1. A parametric query generated dynamically in by the query service user interface

2.2 Graded Level of Query Privacy

The constants which figure in the customer query may need to be protected from a) the service provider and the data sources, b) only from the service provider or c) none. We call them highly sensitive (HS), sensitive (S) and non-sensitive (NS) constants respectively. Each query constant type has defined protection level (Table 2). The customer or the querier, being the owner of the constants, is given the authority to identify each constant as one of the three (preferences [13, 15]). In specific cases however, the identification could be predefined by the system depending on its severity. For example, when searching crime records a criminal's name should belong to HS. In any query specific situation, depending on the number of elements of HS, S and NS variety, we have considered four levels of query privacy (Table 3) for the proposed service. NS constant type provides increased efficiency as they are visible to everyone. No privacy mechanism needs to be built for them. Both S and NS constant types are visible to the databases. This adds to the interests of different data sources to participate in the query service. HS constant type primarily protects the interests of the customer, who is reluctant to reveal them even to the (unknown) databases. In each case there is no compromise on data privacy and result privacy. We have suggested separate algorithms for each case considering the aspects of both privacy and efficiency. In each case first the single-database model has been addressed and then generalized to the multi-database model.

Table 2. Classification of Query Constants

Query Constant	Protected from	Visible to	
		Service Provider	Data sources
Non-sensitive (NS)	None	Yes	Yes
Sensitive (S)	Only service provider	No	Yes
Highly sensitive (HS)	Service provider and the data sources	No	No

Table 3. Graded Level of Privacy

Number of			Query privacy type
NS	S	HS	
≥ 0	0	0	*Public Query:* This type of query has only non-sensitive constants or has no constant at all. These constants, if any, are exposed to the service provider as well as to the data owners.
≥ 0	> 0	0	*Private Query with Full Disclosure of constants to Data Owners*: Apart from non-sensitive constants this type of query has sensitive constants too. All the constants are exposed to the data owners, but the service provider gets exposure of the non-sensitive constants only.
≥ 0	> 0	> 0	*Private Query with Partial Disclosure of constants to Data Owners*: Apart from non-sensitive and sensitive constants this type of query also has highly sensitive constants. The non-sensitive and sensitive constants are exposed to the data owners but the service provider gets exposure of the non-sensitive constants only. The highly sensitive constants are hidden from the data owners.
≥ 0	0	> 0	*Private Query with Non Disclosure of constants to Data Owners:* This type of query has non-sensitive and highly sensitive constants. The non-sensitive constants are dealt as in public query but the highly sensitive constants are not disclosed to the data owners.

2.3 Approach to Maintain Query Privacy

The query constants are contained in the predicates of an SQL query. Our strategies to maintain the privacy of three types of query constants are as follows. The detailed procedure is discussed in Section 4.

a) The non-sensitive constants (NS) do not need any privacy from the service provider and the data owners and therefore they can be directly substituted in the query text by the application program.
b) The sensitive constants (S) need protection from the service provider but they can be revealed to the data owners. Elements of S are sent to the data owners (databases) via service provider in encrypted form. The service provider acts as conduit in this message transfer. The data owners in turn decrypt the constants with the help of the customer via service provider used as a conduit. This ensures that the service provider does not learn the query constants. After obtaining S each database substitutes these values in the respective placeholders and executes its local query. We have proposed *commutative encryption* in this secret message transfer.
c) The highly sensitive constants (HS) need protection from the service provider as well as the data owners. To hide the highly sensitive constants, the application program first computes a public sub query which is basically the original query that has been stripped off the predicates containing the highly sensitive constants. Any attribute lost in this process from the predicate is added to the SELECT list of attributes if not already present. This ensures that the result of the original query is contained within the result of the sub query. This idea has been adopted from Olumofin and Goldberg [9]. The sub query is sent to each data owner through the service provider. Each data owner executes the sub query and sends the result set in encrypted form to the service provider. The service provider in turn selects the tuples of interest of the customer's query from the result sets obtained from each database. For selection of the tuples the elements of HS are sent to the service provider in encrypted form. The HS constants are not seen by the databases, they are only seen by the service provider in encrypted form.

2.4 Query Processing Framework

The proposed query service has the following entities – a service provider SP, a customer C and a number of independent and heterogeneous databases $D_1, \ldots, D_n$ of n data sources. We assume that the data sources cannot communicate with each other and SP itself is not a data source. SP receives customer's query Q and solves the query with the help of a set of databases. C does not have any knowledge about the databases that can resolve the given query. The data sources registered with the SP share their data catalogue for the part of their data they like to share for the query service. With the help of the catalogues SP locates the relevant databases by comparing the schema of the databases with the attributes in the target list and the predicate of the query. Alternatively, SP can also send the formatted query to a set of potential data sources based on the catalogs available or prior knowledge about them. The data sources would then match their database schemas with Q and examine the constants NS, S and HS (without looking into the constant values) with its business objective and decide to participate in the query resolution. Finally SP reformulates and splits the query into local queries and sends to the relevant databases for processing. Each database processes its local query. The query processing engine at each database generates results in accordance with a common schema and sends the result in encrypted form to SP. With the help of SP and the databases, C combines the partial query results to obtain the final query result. The idea is to perform the entire operation in a privacy preserving manner so that SP does not learn anything about the query constants S and HS, about the contents of the databases, and about the results (except possibly its sizes). No database learns anything about the contents of other databases and also others' results and identities of C and other data sources. C does not learn anything about the contents of the databases or identity of the corresponding data source. It only learns the overall result R or individual results R_i, without being able to locate the data source.

III. RELATED WORKS

Aggarwal *et al.* [1] proposed a two-party storage model to enable secure database query service for outsourced data. The key idea in their approach is to partition data into two logically independent database systems according to the privacy constraint; these databases cannot communicate with each other yet execute database query in that distributed architecture. Their proposed scheme does not allow queries to execute on multiple databases. Agrawal, Evfimievski, and Srikant [2] developed protocols for secure intersection, intersection size and equijoin database operations for two databases using commutative encryption and hashing. Their work exposed partial information such as table sizes and the query to the databases and does not support aggregation queries. In [5] Chow, Lee and Subramanian proposed a two-party computation model for privacy-preserving queries over distributed database in an honest but curious adversarial model comprising of two semi honest parties – *randomizer* and *computing engine* other than the customer and the databases. Scalability of query computation over large databases was their focus area, though the proposed model does not support comparison across databases. Their work assumes that randomizer picks up a random string and sends to the databases and the customer, for de-randomization via *confidential channels*. Their model supports data privacy and result privacy but does not consider query privacy. Emekci, Agrawal, Abbadi and Gulbeden [6] proposed privacy preserving intersection, equijoin and aggregation query solution over hash-based P2P system in which selected third parties perform query computation to speed up query response and preserving privacy of the data sources. These third parties are selected from a *peer-to-peer (P2P)* system, namely Chord [12] for computation of the query results. The secrets are distributed to the third parties using Shamir's secret sharing method [10]. Their model considers data privacy but not query privacy. Most of the existing privacy preserving query processing solutions deals with data and query privacy separately. However, in a recent work Hu, Xu, Ren and Choi [7] dealt with data, query and result privacy. Their model is for single database. They used homomorphic encryption, a computationally heavy encryption algorithm to compute Euclidean distance for distance based queries such as kNN query and distance range query. Moreover homomorphic encryption enforces some restrictions on the domain of plaintext. There is a number of privacy preserving techniques for query processing over distributed databases [5, 6, 7, 9]. From traditional database PIR the shift was towards keyword based data retrieval from database. This notion was introduced by Chor, Giloba and Naor [20]. Olumofin and Goldberg [9] showed how to retrieve data from relational database with PIR hiding sensitive constants contained in the predicates of a query. To hide the sensitive constants, the customer computes a public subquery which is basically the original query that has been stripped off the predicates containing sensitive constants. The desired result of the original query is contained within the result of this subquery from which customer retrieves the tuples of interest using PIR by keywords. The work is very similar to the work of Reardon et al. [21] who extended PIR systems to private evaluation of SQL like queries over relational databases. They added capabilities of querying relational databases to traditional PIR techniques. Chor et al. [22] introduced the problem of retrieving information privately from a database; problem of private block retrieval was studied by them. In [20] Chor et al. explored private access of databases by keywords. They combined PIR scheme with data structures that support search operations. The database inserts a set of key values $\{s_1, s_2, \dots, s_n\}$ into a data structure. The customer privately searches this data structure to find keyword of interest; the database learns no information about the keyword being searched. Searching involves *oblivious walk* over the data structure. Performing range queries on encrypted data in the public-key setting was studied in [23]. The work of Olumofin and Goldberg [9] allowed the customer to retrieve data from a database by performing keyword based PIR on the result dataset of the public subquery executed by server after receiving from the customer. This has a partial disclosure issue but the authors claim this to be a practical solution. The work of Reardon et al. [21] however did not allow this to happen. They allowed the customer to parse, optimize and execute database query on the data blocks received from the server by PIR. Advantage is that the database does not learn the textual part of the query but the disadvantage is that of poor performance and the SQL operations being transferred to the customer; thus unable to use

database query optimization and interoperability with existing database systems. The extension is possibly from traditional PIR which assumes that the customer knows the particular block of interest. The concept used in the work of Olumofin and Goldberg [9] has been used in the algorithm for Private Query with No Disclosure. One important benefit of this variant is that it provides the capability of making range searches which is not possible in the other solution as it encrypts the sensitive constants, which does not allow range searches. Query executions in a collaborative cloud [25] in which different parties need to release information and cooperate with others require protection of sensitive information. The data source participating in such systems could be completely independent, federated or a centrally planned distributed database system. Yoon et al. [26] presented a mathematical model for dynamic collaboration of cloud service providers for auction market to offer collaborative services to its customers. In [24] the authors have investigated the problem of collaboration in the context of handling patent applications which involves highly collaborative aspects throughout the stages of information seeking and retrieval. In [27] the authors use role mining to optimize the collaboration within a search session. They use role mining methodology, a data mining task which learns how the participants in collaboration are different and use that knowledge to suggest roles of new participants. Companies that collaborate effectively across the supply chain have enjoyed dramatic reductions in inventories and costs, together with improvements in speed, service levels, and customer satisfaction. Zhao et al. [28] proposed a collaborative query planning service that enables multiple parties to collaboratively plan queries and controls sensitive information disclosure at the same time. The collaborating parties jointly plan query execution in a database network without depending on any trusted third party. In another recent work [29] the authors have proposed a privacy preserving Business Process Recommendation system based on existing business processes of collaborative organizations.

IV. PROPOSED SCHEME

4.1 Query Transformation Phase

In the query service customer's query Q can have three types of constants {NS, S, HS}. Privacy requirement of each type of constant has been listed in Table 2. Implementation approach of privacies of each type of constants has been discussed in Section 2.2. Before presenting the algorithm we explain the methodology of query transformation with help of a query example (Fig 2).

```
SELECT col1, col2 FROM table1
      WHERE col1 <= NS1  AND col2 BETWEEN S1 and S2 AND col3 = HS1
```

Fig 2. Parametric Query with three types of sensitive constants

NS_1 is a non-sensitive constant, S_1 and S_2 are sensitive and HS_1 is a highly sensitive constant.
The above query in parametric form generated by the client application program looks like:

SELECT col1, col2 FROM table1 WHERE col1 <= ? AND col2 BETWEEN ? and ? AND col3 = ?

Let us assume that the constants entered by the customer through the query service's web interface are as follows: $NS_1 = 70$, $S_1 = 1000$, $S_2 = 2000$ and HS_1 = "ABC Limited"
Before sending to SP, the query text goes through the following transformations:

1. The value of non-sensitive constant NS_1 is substituted in the query text as this can be made public as per customer's choice
2. The attribute corresponding to the highly sensitive constant HS_1 is stripped off from the query text and the attribute is added to the select list so that the answer of the original query is contained within the query result of the transformed query

The modified query Q' after above transformations looks like

```
SELECT col1, col2, col3 FROM table1 WHERE col1 <= 70 and col2 BETWEEN ? and ?
```

In the next two sections we present the privacy preserving protocols to obtain the query result.

4.2Privacy Preserving Message Passing Protocols

This section presents three secured message transfer protocols which are used in the privacy preserving query service protocol presented in section 4.3.

Suppose Alice wants to exchange a secret message m with Bob via Carol in a situation where Alice does not know Bob's identity and cannot directly communicate with him but Carol is a third party intermediary who knows both Alice and Bob. Assuming Alice, Bob and Carol are all semi-honest (honest but curious) property of commutative encryption can be exploited in this message transfer.

Commutative Encryption

Let **M** denote a message space and K denote a key space. An encryption algorithm is commutative if the following Equations hold:

a) For all $\boldsymbol{m} \in \boldsymbol{M}$ and for any $\boldsymbol{k_A}, \boldsymbol{k_B} \in \boldsymbol{K}$

$$E_{k_A}\left(E_{k_B}(m)\right) = E_{k_B}\left(E_{k_A}(m)\right) \qquad (1)$$

b) Given $E_i(.)$, there exists a corresponding decryption function $D_i(.) = E_i^{-1}(.)$. Both $E_i(.)$ and $D_i(.)$ are computable in polynomial time.

c) $\forall\, m_1, m_2 \in M$ such that $m_1 \neq m_2$ and a given n, $\varepsilon < \frac{1}{2^n}$

$$P_r\left[E_{k_A}\left(E_{k_B}(m_1)\right) = E_{k_B}\left(E_{k_A}(m_2)\right)\right] < \varepsilon \qquad (2)$$

From Equation (1) it follows that

$D_{k_A}\left(D_{k_B}\left(E_{k_A}\left(E_{k_B}(m)\right)\right)\right) = D_{k_A}\left(D_{k_B}\left(E_{k_B}\left(E_{k_A}(m)\right)\right)\right) = m$ which means the order of encryption need not be followed for decryption.

Equation (1) ensures that if two plaintext messages are same and they are encrypted twice by two different keys they have the same encrypted values regardless of order of encryption. Equation (2) ensures that with very high probability two different messages do not have same encrypted values. Therefore, we can conclude if two ciphertexts are equal, corresponding plaintexts are also equal. The properties of commutative cryptography have been applied in designing our *privacy preserving message passing protocols* which act as the basic building blocks of the proposed algorithm.

If keys are not shared perhaps RSA [31] is most famous example of commutative encryption. Theoretically any secure encryption scheme that satisfies Equations 1 and 2 can be applied to our protocols.

Following protocols are used to privately exchange secret message between two parties via an intermediary. We assume that the Customer and each database have a key pair for encryption and decryption satisfying Equation 1 and 2. This is similar to RSA where both keys are kept secret with the party.

Privacy Preserving Message Transfer (PPMT) protocol for secured transfer of a *Message* from *Source* to *Destination* using *Intermediary* as a conduit

Parameters: Source C, Destination D, Intermediary SP, Message m
1) C encrypts m with its encryption key and sends $E_C(m)$ to D via SP.
2) D encrypts $E_C(m)$ with its encryption key and sends $E_D(E_C(m))$ to C via SP.
3) C decrypts $E_D(E_C(m))$ to obtain $E_D(m)$ and sends $E_D(m)$ to D via SP.
4) D decrypts $E_D(m)$ to obtain m.

Privacy Preserving Encrypted Message Transfer (PPEMT) protocol for secured transfer of a *Message* in *Destination's encryption* from *Source* to *Intermediary* using *Intermediary* as a conduit

Parameters: Source C, Destination D, Intermediary SP, Message m
1) C encrypts m with its encryption key and sends $E_C(m)$ to D via SP.
2) D encrypts $E_C(m)$ with its encryption key and sends $E_D(E_C(m))$ to C via SP.
3) C decrypts $E_D(E_C(m))$ to obtain $E_D(m)$ and sends $E_D(m)$ to SP.

Privacy Preserving Encrypted Message Transfer (PPEMT) protocol for secured transfer of an Encrypted *Message* from *Source* to *Destination* for decryption by *Source* using *Intermediary* as a conduit

Parameters: Source D, Destination C, Intermediary SP, Message m

1) C receives $E_D(m)$ from D, encrypts with its encryption key and sends $E_C(E_D(m))$ to D via SP.
2) D decrypts $E_C(E_D(m))$ with its decryption key and sends $E_C(m)$ to C via SP.
3) C decrypts $E_C(m)$ to obtain m.

4.3 Privacy Preserving Query Service Protocol

1) Customer C fires a query through the web interface to SP.
2) SP chooses a set of databases D_1, ..., D_n to process Customer C's query Q.
3) C transforms Q into Q' [by client side program at C] and sends to SP // query transformation
4) SP sends Q' to each D_i.
5) C sends set of sensitive constants S to each D_i following *PPMT(C,D_i,SP,S)*.
6) Each D_i executes Q' to generate result set R_i, encrypts it with its encryption key and sends encrypted R_i to SP for further processing.
7) C sends the set of highly sensitive constants HS to SP for each D_i following *PPEMT(C,D_i,SP,HS)*
8) On receipt of the query result R_i from each data source, SP picks the tuples of interest by (equality) matching the encrypted data $E(R_i)$ with each s in E(HS) and sends the resultant data $E(r_i)$ to C.
9) For each data source D_i , C decrypts $E(r_i)$ using *PPEMT(D_i,C, SP,$E_{Di}(r_i)$)*
10) C collates the result sets r_is to get the final answer.

Procedure stated above caters to the case where all the constant sets are nonempty. However each variation (Table 3) will call for suitable modification of the above algorithm. For example, if HS is empty, the databases need not send the partial result to SP for tuple selection. Instead each database can send its local query result to C – *PPMT(D,C,SP, m)*. On the other hand if the query is public simple substitution of the NS constants during transformation process will suffice.

4.4 Security Analysis

Our query processing framework is based on a *semi-honest* or *honest but curious model* which means that all the parties follow the protocol correctly but they may record any intermediate input received during the protocol execution and try to derive some benefit out of it. Databases are not known to the Customer, from a set of data sources the databases are chosen by the Service Provider based on the query type. Moreover, they do not have any common interest. So practically they have no chance of colluding with the Customer. However being known to the Service Provider the databases may collude with each other. Our attempt is to prevent the collusion between them. We analyze security aspect of each protocol. Query privacy of the Customer and the data privacy of the data sources are the primary consideration in this semi honest model. Service provider remains completely in dark during the process but provides the service without learning the query, data and the result of computation.

Any party - the Customer, the Service Provider or the data source may act as an adversary. The encryption/decryption algorithms are known to all the parties but the secret key pair of each party is unknown to others. Following situations may arise:

a. The service provider may like to know the value of the constants of the parametric query given by the customer, the data values of the attributes corresponding to the sensitive constants as well as the query result
b. The customer may like to know the data values of the attributes corresponding to the sensitive constants.

c. The data source may like to know the values of the sensitive constants and other data sources' data.

Security of data depends on the security of the encryption/decryption key pair of individual players.

V. CONCLUSION

The problem of preserving privacy of sensitive constants in customer's query in a query service framework has been studied in this paper. In this model a service provider provides query service with the help of different data sources virtually forming a distributed database with horizontal partitioning, though the data sources could be heterogeneous. Query processing in this model preserves data privacy of the data sources and the query privacy of customer. It also preserves identity privacy of the customer and the data sources and the result privacy. An important tool in our work is *commutative encryption* for secure data exchange between any two parties via a third party. We can use a variant of RSA one-way accumulator also known as exponential accumulator originally advocated by Benaloh and de Mare [3]. The variant was suggested by Kantarcioglu and Clifton [8]. One-way accumulator has commutative property. One-way accumulator has been proved to be computational indistinguishable [17] which makes it a proper choice of encryption mechanism.

To leverage the current momentum of cloud computing and cloud based services and collaborative computing, the proposed privacy aware query service can find its place as a brokerage service provided by a cloud service provider [14]. The query service can extend beyond text to include multimedia like audio, image, video [16]. As more and more people are using cloud services and big players like Amazon, Google are the providers, cloud is a natural destination for this service. Future work will include privacy preserving vertical integration of data and offer a complete solution for a general query processing steps involving a sequence of horizontal and vertical data integration from multiple data sources.

REFERENCES

[1] Aggarwal G., M. Bawa, P. Ganesan, H. Garcia-Molina, K. Kenthapadi, R. Motwani, U. Srivastava, D. Thomas, Y. Xu. "Two can keep a secret: A distributed architecture for secure database services" In Proc. of CIDR 2005.
[2] Agrawal R, A. Evfimievski, and R. Srikant. "Information sharing across private databases". In Proc. of the 2003 ACM SIGMOD international conference on Management of data, pages 86–97, 2003.
[3] Benaloh J. and M. de Mare. "One-way accumulators: A decentralized alternative to digital signatures". In EUROCRYPT '93: Workshop on the theory and application of cryptographic techniques on Advances in cryptology, pages 274–285. Springer-Verlag New York, Inc., 1994.
[4] Boss G., P.Malladi, D. Quan, L. Legregni, H. Hall - IBM Corporation 2007, Service provider computing.
[5] Chow S. S. M., J.H. Lee, and L. Subramanian, "Two-party computation model for privacy-preserving queries over distributed databases," in NDSS, 2009.
[6] Emekci F., D. Agrawal, A. E. Abbadi, and A. Gulbeden. "Privacy preserving query processing using third parties". in *ICDE 2006*, page 27. IEEE Computer Society, 2006.
[7] Hu H., J. Xu, C. Ren and B. Choi. "Processing private queries over untrusted data service provider through privacy homomorphism", In: *ICDE* IEEE Computer Society (2011), p. 601-612.
[8] Kantarcioglu M. and C. Clifton. "Privacy-preserving distributed mining of association rules on horizontally partitioned data". In The ACM SIGMOD Workshop on Research Issues on Data Mining and Knowledge Discovery (DMKD'02), pages 24-31, June 2 2002.
[9] Olumofin F. and I. Goldberg. "Privacy-preserving queries over relational databases". In PETS'10, Berlin, 2010.
[10] Shamir A., "How to share a secret," Commun. ACM, vol. 22, no. 11, pp. 612–613, 1979.
[11] Silberschatz A., H. F. Korth, and S. Sudarshan. Database System Concepts. McGraw-Hill, Inc., New York, NY, USA, 5th edition, 2005.
[12] Stoica I., R. Morris, D. R. Karger, M. F. Kaashoek, and H. Balakrishnan. "Chord: A scalable peer-to-peer lookup service for internet applications". In SIGCOMM 2001, pages 149–160, 2001.
[13] Pearson, Siani, Yun Shen, and Miranda Mowbray. "A privacy manager for cloud computing." *Cloud Computing*. Springer Berlin Heidelberg, 2009. 90-106
[14] What's a cloud services broker, and why do you need one? By Chris Carroll Aug 2014 http://www.cio.com/article/2462417/cloud-computing/cloud-computing-whats-a-cloud-services-broker-and-why-do-you-need-one.html

[15] Henze, Martin, et al. "User-driven Privacy Enforcement for Cloud-based Services in the Internet of Things." *2014 International Conference on Future Internet of Things and Cloud (FiCloud)*. 2014.
[16] Negoiță, Cătălina, and Monica Vlădoiu. "Querying and Information Retrieval in Multimedia Databases."
[17] de Meer, H., Liedel, M., Pöhls, H. C., Posegga, J., & Samelin, K. (2012).*Indistinguishability of one-way accumulators*. Technical Report MIP-1210, Faculty of Computer Science and Mathematics (FIM), University of Passau.
[18] Pal, A. K., S. Bose, "Information Retrieval as a Service for Multiple Heterogeneous Data-Privacy Model", in B.H.V. Topping, P. Iványi, (Editors), "Proc. 3rd Int. Conf. Parallel, Distributed, Grid and Cloud Computing for Engineering", Civil-Comp Press, Stirlingshire, UK, Paper 31, 2013. doi:10.4203/ccp.101.31
[19] Pal, A.K., S. Bose, Information Retrieval as a Service - IRaaS: A Concept Paper on Privacy Analysis, WPS 763, Indian Institute Management Calcutta, June 2015. https://facultylive.iimcal.ac.in/sites/facultylive.iimcal.ac.in/files/WPS%20763.pdf
[20] Chor B., N. Gilboa, and M. Naor. "Private information retrieval by keywords". Technical Report TR CS0917, Dept. of Computer Science, Technion, Israel, 1997.
[21] Reardon J., J. Pound, and I. Goldberg. "Relational-Complete Private Information Retrieval". Technical report, CACR 2007-34, University of Waterloo, 2007.
[22] Chor B., O. Goldreich, E. Kushilevitz, and M. Sudan. "Private information retrieval" Proc. of IEEE Symposium on Foundations of Computer Science, Milwaukee, WI USA, October 23-25 1995.
[23] Boneh D. and B. Waters. Conjunctive, subset, and range queries on encrypted data. In TCC '07, pp. 535-554. Springer, 2007.
[24] Hansen, Preben, Chirag Shah, and Claus-Peter Klas, eds. Collaborative Information Seeking: Best Practices, New Domains and New Thoughts. Springer, 2015.
[25] De Capitani di Vimercati, S. Foresti, P. Samarati. "Managing and accessing data in the cloud: Privacy risks and approaches", Int. Conf. *Risk and Security of Internet and Systems (CRiSIS)*, IEEE, 2012.
[26] Yoon, C., M. M. Hassan, H. Lee, W. Ryu, E.N. Huh, "Dynamic collaborative cloud service platform: opportunities and challenges." *ETRI j.* 32.4 (2010).
[27] S. Laure, C. Shah, and L. Tamine. "User-driven system-mediated collaborative information retrieval."In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*, pp. 485-494.ACM, 2014.
[28] Zhao, Mingyi, Peng Liu, and Jorge Lobo. "Towards Collaborative Query Planning in Multi-party Database Networks." *Data and Applications Security and Privacy XXIX*. Springer International Publishing, 2015. 19-34.
[29] Irshad, H., Shafiq, B., Vaidya, J., Bashir, M. A., Shamail, S., & Adam, N. Preserving Privacy in Collaborative Business Process Composition.
[30] Khayat, Saied Hosseini. "Using commutative encryption to share a secret."*Electrical Engineering Department, Ferdowsi University of Mashhad, Iran*(2008).
[31] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," Communications of the ACM, vol. 21, no. 2, pp. 120–126, 1978. [Online]. Available: http://doi.acm.org/10.1145/359340.359342