Asian Journal of Convergence in Technology Volume IV Issue II
ISSN NO: 2350-1146 I.F-5.11

Querying methods of Encrypted Cloud Data

[1]Vemula Sridhar, [2]Marneni Dyna
Department of CSE, MVSR Engineering College, Hyderabad.
Email: [1]sridhar_cse@mvsrec.edu.in, [2]dyna_cse@mvsrec.edu.in

***Abstract*—In the recent times business organizations are opting for cloud based storage for reducing maintenance and storage cost for which data security is a major issue of concern. The Organizations may not completely rely on security provided by cloud service provider. Instead they would prefer for their own security model. Here data may be in relational or Non-relational with its own structures. If the data stored in the cloud is in encrypted format, querying will be difficult because for every retrieval process, the data has to be decrypted. This is again security problem with server. This paper explores about how to process and query the encrypted data stored in cloud. Different methods of querying, ranging from relational to Non-relational data are discussed in this paper.**

Keywords—querying, cloud encryption, cloud security, information retireival

I. Introduction

Computing technology is moving towards cloud computing technology in various domains. As such the primary need of any organization is to maintain enormous amount of user data to retain the users for a longer period. This can be achieved by incorporating high capacity storage servers but it is expensive when compared to the cloud storage.

Cloud computing, the new trend in Information Technology world provides users with great flexibility, based on pay-per-usage pricing model. This enables the organizations to incorporate their business instantly thus reducing the cost of time and money to purchase new hardware and software.

Many cloud storage providers are available today, such as Amazon Simple Storage Service (S3), Google drive, one drive etc. Consequently IT users and business organization are moving their data towards cloud to reduce their infrastructural and maintenance cost for storage.

Data can be outsourced to cloud storage so that instead of storing it in local , high cost storage servers, we can store it with cloud service provider and data can be retrieved whenever needed. However as we are moving data from local servers to public servers, data security becomes major issue. Leakage of client confidential information can cause big problem to the organization, hence special care has to be taken to provide security to confidential information.

Some cloud storage service providers are coming with built-in encryption support but the client cannot completely rely on the cloud provider in terms of confidentiality of user data. One choice here is to encrypt and upload data from the client itself. Both encryption and effective retrieval of data from the cloud have to be managed.

The primary concern of the client is to minimize the maintenance cost by moving all the data towards the cloud without compromising on security or privacy of data and at the same time have an effective retrieval and querying. This demands a challenge from the perspective of cloud computing.

A great deal of research has been focused on several aspects of security in cloud. To preserve privacy and provide security to data stored in the cloud, data encryption methods are more commonly being used. Cloud storage service providers can provide encryption feature to safeguard the confidential data, using strong and efficient cryptographic algorithms.

Various public and private encryption algorithms (AES, RSA etc..) are available which provide strong security to data stored in cloud storage but once data is encrypted using these algorithms, retrieval and querying may not be possible without decryption because querying methods are not operated on direct data instead they operates on encrypted data.

This problem can be solved by decrypting the data before every query operation, operate the query process then results must be encrypted again to send it to client over network. But if the organization has some data that is very sensitive, it is not recommended to completely rely on cloud storage provider in terms of data security.

It is possible to choose such encrypted cloud storage providers if organization is only storing data without retrieval each time or if the company's data is not very sensitive [10].
However if an organization needs to retrieve and query data from encrypted cloud, it is a big challenge as the data is

encrypted. So it is difficult to operate query and retrieval methods on encrypted data and get the required results.

One solution is encrypting the data from client even before uploading them on to the cloud storage so that the data will not be prone to attackers including, from the service provider. Encrypted data storage though, safeguards the data from attacks; it creates problems when data has to be retrieved frequently by the client. Hence there has been research in the direction of encryption methods which allow retrieval of encrypted data without the need for decrypting it.

In recent trends solutions are available to operate queries on encrypted data but they are not in complete implementation due to some limitations related to performance and reliability. Advances in the research are making use of powerful encryption scheme to encrypt the cloud data and at the same time would provide faster querying.

The rest of the paper is organized as follows: In Section-2, we discuss various special encryption methods used in querying encrypted data. In Section-3, we introduce CryptDB software developed using these encryption methods and Mylar-a Web based framework for providing security.

II. SPECIAL ENCRYPTION METHODS

There are many encryption methods that are available .For example (DES, AES, RSA etc...), protect the data that is stored remotely. These methods convert plain text into different form of cipher text, in which the order and relationship that existed in the plain text is not maintained. Therefore encryption methods that allow processing of data without decryption are needed. At the same time security should not be compromised.

In information retrieval data can be in relational form or non-relational form. If the data is relational then different relational database management systems are available to store the data. Here information from database will be retrieved by sending queries to database. Various types of queries include getting record information given primary key value. It is also possible to retrieve set of records which satisfies specific conditions on an attribute value. The encryption method chosen to encrypt relational data should maintain plaintext property in generated cipher text then only querying without decryption is possible. CryptDB [1] is a database where original data is stored in encrypted format so that the administrator or cloud provider does not have any information about plain text except the order. CryptDB will be discussed in further section.

If data is non-relational, then they are characterized by their own structure. It follows structured data format. Here data is stored in the form of documents. Set of documents are uploaded to cloud by administrator. To enable fast retrieval from this data index in constructed and stored on set of documents at cloud. Querying or retrieval on these documents will be done by keyword based search. Given a keyword, server will return top k documents containing given keyword [6]. Top k documents will be generated based on TF-IDF factor, which indicates keyword relevance frequency in documents.

In case of non-relational data the documents are first encrypted and then transferred to cloud server to ensure security. The index generated on these documents is also encrypted before uploading it to cloud server. The encryption algorithm should be chosen such that top k documents for a given keywords can be retrieved without decryption.

In both relational and non-relational data, the order should be retained in cipher text so that comparisons are made possible.

And also it should be able to perform operations other than comparison on cipher text without decryption.

Order Preserving Encryption [2] first proposed by Boldivera retains the plaintext order in generated cipher text. This encryption model is suitable to retrieve data based on comparison because retrieving data without decryption is possible with order comparison.

Homomorphic encryption [3] first proposed by crain gentry is another type of encryption which supports operation on encrypted data to get required result without decrypting original plain text.

In next two sub section these methods are discussed.

A. ORDER PRESERVING ENCRYPTION

It is a symmetric encryption method and it retains the order between data elements even after encryption, so it also called as order-preserving symmetric encryption (OPSE) [4]. The order preserving means for two plain text elements x1,x2 and if there is a relationship between them as $x1<x2$, then after encryption its corresponding cipher texts are E(x1) and E(x2), and they satisfy the condition $E(x1)<E(x2)$. Boldyreva et al. [11] initiated the cryptographic study of OPE schemes, and they defined the security of an OPE scheme using the ideal object. Note that any order-preserving function g from domain $D = \{1, 2, \cdot\ \cdot, M\}$ to range $R = \{1, 2, \cdot\ \cdot, N\}$ can be uniquely defined by a combination of M out of N ordered items. The ideal object is just a function that is randomly selected from all order-preserving functions, which is called a random order-preserving function (ROPF). Thus, with the spirit of pseudorandom functions, an OPE scheme is defined to be secure if the adversary cannot

distinguish the OPE from the ROPF. In [4], the authors also constructed an efficient OPE scheme satisfying this secure criterion. The construction is based on the relation between the random order-preserving function and the hyper-geometric probability distribution (HGD), and a HGD simpler is used to select an order-preserving function in a pseudorandom manner.
In the OPE scheme of [4], the range R is divided into some non overlapping interval buckets with random sizes. The random-sized bucket is determined by a binary search based on a random HGD sampler. The procedure of binary search is described as Algorithm 1, where TapeGen () is a random coin generator.

Algorithm 1 Binary Search

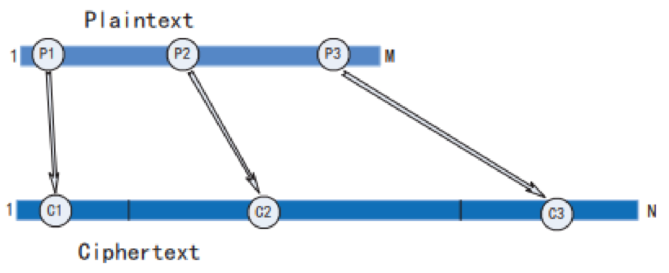
```
Input: {K, D, R, m}
1: M ← length(D); N ← length(R)
2: d ← min(D) − 1; r ← min(R) − 1
3: y ← r + ceil(N/2)
4: coin $\xleftarrow{R}$ TapeGen(K,(D, R, y||0))
5: x $\xleftarrow{R}$ d + HGD(coin, M, N, y − r)
6: x = d + f
7: if m ≤ x then
8: D ← {d + 1, · · · , x}
9: R ← {r + 1, · · · , y}
10: else
11: D ← {x + 1, · · · , d + M}
12: R ← {y + 1, · · · , r + N}
13: end if
Output: {D, R}
```

After the binary search, a plaintext m is mapped into a bucket in the range R, and then the OPE algorithm assigns a fixed value in the bucket as the encrypted value of m. The encryption process of Algorithm 1 is illustrated in Fig.1 (a), which shows that a given plaintext mi will always be mapped to a fixed cipher text c_i belonging to a bucket selected by the binary search procedure; therefore it is a deterministic encryption.

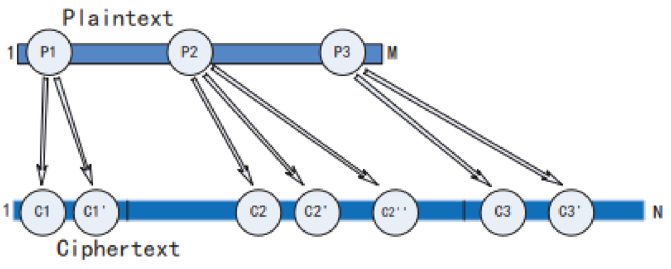
One-to-Many OPE:

In applications of privacy preserving keyword search, if a deterministic OPE is used to encrypt relevance scores, the cipher texts will share exactly the same distribution as its plain counterpart, by which the server can specify the keywords.

Therefore, Wang et al. [4] modified the original OPE [3] to a probabilistic one, called "One-to-Many OPE". For a given plaintext m, i.e., a relevance score, the "One-to-Many OPE" first employs Algorithm 1 to select a bucket for m, and then randomly chooses a value in the bucket as the cipher text. As shown in Fig.1(b) given plain text is mapped to different cipher text bucket for same plaintext. The randomly choosing procedure in the bucket is seeded by the unique file IDs together with the plaintext m, and thus the same relevance score in the Inverted Index will be encrypted.



a) Deterministic OPE



b) One-to-Many OPE

Fig. 1. Comparison between OPE and One-to-Many OPE

B.HOMOMORPHIC ENCRYPTION

Homomorphic encryption [7] allows us to perform operations on encrypted data without knowing the private key and without decrypting that data. When we decrypt the result of any operation, it is the same as if we carried out the calculation on the plain data. In the process of building a system with such capabilities, a plethora of systems were developed that offer partial functionalities.
Those types of Homomorphic encryption systems are known as partially homomorphic encryption systems. In 2009, Craig Gentry in his PhD thesis described the first fully Homomorphic encryption system (FHE) [3]. From then, a number of systems were (and still are) developed that offer similar or better characteristics.

The basic Homomorphic algorithm consists of:

1. Generate public key p_k and secret key s_k.
2. Encrypt function that takes public key p_k and message m and outputs cipher text.
3. Decrypt function that takes cipher text and secret key and outputs message.
4. Evaluate function that takes a function with logical gates and set of cipher texts and outputs cipher text c_e.

Types of Homomorphic systems:

- Fully Homomorphic: they support both addition and multiplication.
- Partially Homomorphic systems: they support single operation on encrypted data.
- There is also another type that allows many operations of one type and limited operations of another type.

A Homomorphic encryption is additive if:
$$Enc(a+b) = enc(a).\ enc(b)$$

A Homomorphic encryption is multiplicative if:
$$Enc(a.b)= enc(a).enc(b).$$

Limitations of Homomorphic encryption:

The cipher text generated contains random noise where addition roughly doubles the noise and multiplication squares the noise. This noise increases further on successive Homomorphic operations. The cipher text with a particular range of noise can only be decrypted correctly. Because of this limitation arbitrary number of operations cannot be performed on Homomorphic encryption.

III. SOFTWARES

Different softwares are developed with above specified encryption methods in different scenarios related to database, web applications etc. There is enough scope of research in various domains to develop softwares and applications by utilizing OPE, Homomorphic Encryption types, so that can be utilized in real time environment. CryptDB- Database which stores and performs operations on encrypted data itself and Mylar web application framework is discussed in following sub sections.

A. CryptDB

As fully Homomorphic encryption systems are very expensive, a more practical approach is provided by CryptDB for security of sensitive data.

CryptDB has designed a system to provide security for applications dealing with large databases. CryptDB protects the system from unauthorized user who tries to access confidential information. It minimizes the amount of sensitive data to the server and efficiently executes the queries. CryptDB works by executing queries on encrypted data in a DB proxy. The proxy encrypts and decrypts data and changes query operators. CryptDB uses SQL aware encryption strategy. According to this strategy, SQL consists of several operators like equality checks, joins, comparisons operators etc. By using known encryption methods for joins, CryptDB encrypts the data so that DBMS executes the transformed queries. Figure 2 represents the architecture of CryptDB.

CryptDB is efficient as it uses symmetric key encryption. It optimally protects most of the sensitive fields with highly secure encryption schemes. It is not optimal for more revealing encryption schemes but these are often on less sensitive fields eg: timestamps. Leaks at most the data of currently active users for the duration of compromise.

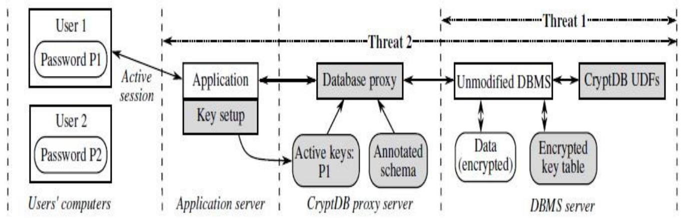


Figure 2. The architecture of CryptDB.

CryptDB architecture includes three levels of functionality.
1. Application server
2. CryptDB proxy server
3. DBMS Server

1. Application server

- runs the application code and issues DBMS queries on behalf of one or more users
- is modified so that it provides the db proxy with encryption keys

2. DBMS server

- all its data is encrypted (including table and column names)
- Processes encrypted data as if it were unencrypted

- has user defined functions (UDFs) installed to allow it to compute on cipher texts for certain operations
- has some auxiliary tables (ex. Encrypted keys) used by the db proxy

3. Database proxy

- Encrypts queries received from application and sends them to DBMS
- Changes some query operators if necessary, while preserving query's semantics
- Decrypts DBMS returned results and send them to the application
- Stores master key(s) and an annotated version of application schema (which it uses to check access rights, and to keep track of current encryption level, or onion layer, on each column)
- Decides the key = f(user, query) to be used for encrypting/decrypting each data item

Different steps in Processing of a given query by CryptDB:

1. Db proxy intercepts application's query and rewrites it by anonymizing table and column names & encrypting constants with the key of the encryption scheme best suited for the operation and the user
2. Db proxy checks if the DBMS needs to adjust encryption level before executing the query. If yes, issue an UPDATE query that invokes a UDF to adjust the encryption level layer of the appropriate columns
3. Db proxy sends the encrypted query to the DBMS server
4. DBMS executes query using standard SQL (invoking UDFs for aggregation or keyword searches) and returns the encrypted results
5. Db proxy intercepts and decrypts results, and sends them to the application

B. Mylar

Mylar [8] is a new platform for building web applications that stores only encrypted data on the server. Mylar makes it practical for many classes of applications to protect confidential data from compromised servers. It leverages the recent shift in web application frameworks towards implementing logic in client-side JavaScript code, and sending data, rather than HTML, over the network [5]; such a framework provides a clean foundation for security. Mylar addresses the security challenges with a combination of systems techniques and novel cryptographic primitives, as follows

- Data Sharing
- Computing over encrypted data
- Verifying application code

It addresses the following:

1. Provides platform for storing encrypted data on the servers with complete access.
2. Provides sharing of data among several authenticated users.
3. Mylar checks that the client side code is not tampered.

Several sites perform encryption and decryption at the browser before sending it to the server. But the server itself may be untrusted. Mylar provides web application security with browser extension that verifies that the client side application code is authentic, even if the server is malicious. Mylar makes use of client-side library to intercept data sent to and from the server and encrypts or decrypts that data. The client-side library stores the private key of the user at the server, encrypted with the user's password. When the user logs in, the client-side library fetches and decrypts the user's private key. For shared data, it creates separate keys at the server in the encrypted form.

In most web applications, different users access data and hence make use of different keys for encryption. Mylar provides cryptographic method that enables keyword search mechanism in web applications over data encrypted with different keys. Server side library performs computation on the encrypted data at the server side to support keyword search.

For some applications Mylar makes use of IDP (identity provider) to check that a given public key belongs to a particular username. An application uses the IDP if it is unaware of the users and IDP enables the application to choose the users for data sharing.

Threats from Mylar

1. Mylar checks that a particular user is authenticated but does not ensure that the results from a computation at the server are correct or not.
2. Mylar assumes that a user checks the browser security indicator. If the user fails to check then Mylar does not guarantee for the data confidentiality.

IV. CONCLUSION

The survey given in this paper discussed about different ways of processing encrypted cloud data. For structured and un-structured encrypted data it is possible to retrieve information without decryption using special encrypted methods and CryptDB discussed above. Some of them are not completely in public use but they are open for research scope.

REFERENCES

[1] Raluca Ada Popa, Catherine M. S. Redfield, Nickolai Zeldovich, and Hari Balakrishnan. 2011. CryptDB: protecting confidentiality with encrypted query processing. In Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles (SOSP '11). ACM, New York, NY, USA, 85-100, DOI=http://dx.doi.org/ 10.1145/2043556.2043566

[2] A. Boldyreva, N. Chenette and Y. Lee , “Order-preserving symmetric encryption,” Advances in Cryptology-EUROCRYPT, 2009. Springer Berlin Heidelberg, pp. 224-241, 2009.

[3] C.Gentry, “A fully homomorphic encryption scheme,” Ph.D. dissertation, Stanford, CA, USA, 2009

[4] A. Boldyreva, N. Chenette, Y. Lee, A. O’Neill, Order-preserving symmetric encryption, Eurocrypt 2009, pp. 224-241.

[5] Meteor, Inc. Meteor: A better way to build apps. http://www.meteor.com, Sept. 2013

[6] Ning Cao, Cong Wang, Ming Li, Kui Ren, Wenjing Lou, Privacy-Preserving Multi-Keyword Ranked Search over Encrypted Cloud Data, IEEE Transactions on Parallel & Distributed Systems, vol.25, no. 1, pp. 222-233, Jan. 2014

[7] Hrestak, D.; Picek, S., Homomorphic encryption in the cloud Information and Communication Technology, Electronics and Microelectronics (MIPRO), 2014 37th International Convention on, vol., no., pp.1400, 1404, 26-30 May 2014

[8] Raluca Ada Popa, Emily Stark, Jonas Helfer, Steven Valdez, Nickolai Zeldovich, M. Frans Kaashoek, and Hari Balakrishnan. 2014. Building web applications on top of encrypted data using Mylar. In Proceedings of the 11th USENIX Conference on Networked Systems Design and Implementation(NSDI'14). USENIX Association, Berkeley, CA, USA, 157-172

[9] Alexandra Boldyreva, Nathan Chenette, and Adam O'Neill. 2011. Order-preserving encryption revisited: improved security analysis and alternative solutions. In Proceedings of the 31st annual conference on Advances in cryptology (CRYPTO'11), Phillip Rogaway (Ed.). Springer-Verlag, Berlin, Heidelberg, 578-595

[10] Hingwe, K.K.; Bhanu, S.M.S., "Two layered protection for sensitive data in cloud," Advances in Computing, Communications and Informatics (ICACCI, 2014 International Conference on, vol., no., pp.1265, 1272, 24-27 Sept. 2014