International Journal of Recent Trends in Engineering & Research
ISSN(ONLINE) : 2455-1457
IMPACT FACTOR : 3.344

Advanced Querying And Information Retrieval

Ms. M.SANDHYAI[1], Mr. D.ANIL[2],
[1]*M.tech 2nd year,Dept. of CSE, VEC, Kavali, India*
[2]*Assosiate professor,Dept. of CSE, VEC, Kavali, India*

ABSTRACT- The most challenging research works in computing is privacy and protection. it provides an innovative business model for organizations with minimal investment. security is one of the major issues in cloud computing. currently huge researches is taking place on cloud computing. this is a survey paper on query services in cost efficient clouds .generally user expects immediate result for any query .but it is a cost efficient even we may adjust with the dealy in query services .this survey presents description and comparision of Ostrovsky, COPS, and EIRQ protocols. EIRQ protocol is the latest among these protocols and it address the issues of privacy, aggregation, CPU consumption and network bandwidth usage.

I. INTRODUCTION

Cloud computing is the internet based storage method. It is mainly used for storing the files and applications infrastructures .Peoples uses the cloud because of its attractive features like secure service, infinite of storage, it will satisfy the user experience, low cost multiple user can access the files and applications. In cloud, the query service process are frequently used because, the user can save their cost. The owners in the cloud will pay the amount only for their using time of server. This is a important feature because; the working time of query service in cloud is very high and more expensive. Cloud application, an organization subscribes the cloud services and authorizes its staff to share files in the cloud. Each file is described by a set of keywords, and the staff, as authorized users, can retrieve files of their interests by querying the cloud with certain keywords. In such an environment, how to protect user privacy from the cloud, which is a third party outside the security boundary of the organization, becomes a key problem. Private searching was proposed by Ostrovsky .which allows a user to retrieve files of interest from an untrusted server without leaking any information. However, the Ostrovsky scheme has a high computational cost, since it requires the cloud to process the query on every file in a collection. Otherwise, the cloud will learn that certain files, without processing, are of no interest to the user. It will quickly become a performance bottleneck when the cloud needs to process thousands of queries over a collection of hundreds of thousands of files.

We argue that subsequently proposed Commercial clouds follow a pay-as-you-go model, where the customer is billed for different operations such as bandwidth, CPU time, and so on. Solutions that incur excessive computation and communication costs are unacceptable to customers.To make private searching applicable in a cloud environment, our previous work designed a cooperate private searching protocol (COPS), where a proxy server, called the aggregation and distribution layer (ADL), is introduced between the users and the cloud. The ADL deployed inside an organization has two main functionalities: aggregating user queries and distributing search results. Under the ADL, the computation cost incurred on the cloud can be largely reduced, since the cloud only needs to execute a combined query once, no matter how many users are executing queries.

II. LITERATURE SURVEY

Literature Survey analysis the project concept with the standard papers and journal. We can understand the methodology that follows and implementing some of new idea. To analysis the papers main goal and find the possible way to applying in the different environment Motivated by this goal, we propose a scheme, termed Efficient Information retrieval for Ranked Query (EIRQ), in which each user can choose the rank of his query to determine the percentage of matched files to be returned. The basic idea of EIRQ is to construct a privacy-preserving mask matrix that allows the cloud to filter out a certain percentage of matched files before returning to the ADL. This is not a trivial work, since the cloud needs to correctly filter out files according to the rank of queries without knowing anything about user privacy. Focusing on different design goals, we provide two extensions: the first extension emphasizes simplicity by requiring the least amount of modifications from the Ostrovsky scheme, and the second extension emphasizes privacy by leaking the least amount of information to the cloud .We propose three EIRQ schemes based on the ADL to provide a cost-efficient solution for private searching in cloud computing..The EIRQ schemes can protect user privacy while providing a differential query service that allows each user to retrieve matched files on demand.

III. EXISTING SYSTEM

Our work aims to provide differential query services while protecting user privacy from the cloud. Existing research that is similar to ours can be found in the areas of private searching Unlike searchable encryption where the user conducts searches on encrypted data, private search in performs keyword-based searches on unencrypted data. Private searching was first proposed in which allows a server to filter streaming data without compromising user privacy. Their solution requires the server to return a buffer of size $O\delta f \log(f)$ when f files match a user's query. Each file is associated with a survival rate, which denotes the probability of this file being successfully recovered by the user. Based on the Paillier cryptosystem , the files that mismatch a query will not survive in the buffer, but the matched files enjoy a high survival rate .Among various extensions, [further reduced the communication cost from $O\delta f \log(f)$ to $O\delta f$ þ by solving a set of linear equations to recover f matched files. However, their scheme requires the decryption of one more buffer, thus the computation cost is higher than the Ostrovsky scheme. presented an efficient decoding mechanism which allows the recovery of files that collide in a buffer position. Reference proposed a recursive extraction mechanism, which requires a buffer of size $O\delta f$ þ when f files match a user's query. Reference proposed two new communication-optimal constructions; one uses Reed-Solomon codes and allows for a zero-error, and the other is based on irregular LDPC codes and allows for lower computation cost at the server. The above private searching schemes only support searching for OR of keywords or AND of two sets of keywords. Reference extended the types of queries to support disjunctive normal forms (DNF) of keywords. The main drawback of existing private searching schemes is that both the computation and communication costs grow linearly with the number of users executing queries. Thus, when applying these schemes to a large-scale cloud environ- ment, querying costs will be extensive .Our previous work was the first to make private searching techniques applicable to a cloud environment. However, requires the cloud to return all of the matched files, which may cause a waste of bandwidth when only a small percentage of files are of interest. To alleviate the problem, we introduced the concept of differential query services in . The main difference between this work and is that we provide two extensions to address different aspects of the problem, and we conduct extensive experiments on a real cloud to verify the effectiveness of the proposed schemes.

IV. PROPOSED SYSTEM

This paper provides the survey on different query services in cloud like

- An Efficient Information Retrieval for Ranked Queries (EIRQ) scheme is recovery of ranked files on user demand. An EIRQ work based on the Aggregation and Distribution Layer (ADL). An ADL is act as mediator. The system mainly consists of three entities: the aggregation and distribution layer (ADL), many users, and the cloud, as shown in Fig
- The basic idea of EIQR-Efficient is to construct a privacy-preserving mask matrix with which the cloud can filter out a certain percentage of matched files before mapping them to a buffer. As proven in the Ostrovsky scheme, the file survival rate is determined by the buffer size β and mapping times γ. Therefore, the basic idea of two extensions is that, for each rank I € {0….r} the ADL adjusts the buffer size βi and the mapping times γi to make the file survival rate qi approach 1-i/r.

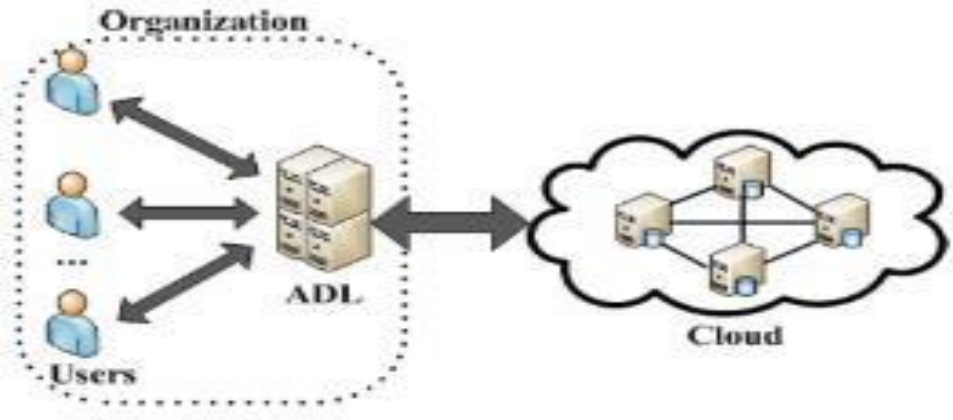

Fig 1: System Model.

- Before illustrating EIQR-Efficient, two fundamental problems should be resolved: Firstly, we should determine the relationship between query rank and the percentage of matched files to be returned. Suppose that queries are classified into 0 - r ranks. Rank-0 queries have the highest rank and Rank - r queries have the lowest rank. In this paper, we simply determine this relationship by allowing Rank-i queries to retrieve (1-i/r) percent of matched files. Therefore, Rank-0 queries can retrieve 100 percent of matched files, and Rank-r queries cannot retrieve any files.

Secondly, we should determine which matched files will be returned and which will not. In this paper, we simply determine the probability of a file being returned by the highest rank of queries matching this file. Specifically, we first rank each keyword by the highest rank of queries choosing it, and then rank each file by the highest rank of its keywords. If the file rank is i, then the probability of being filtered out is i=r. Therefore, Rank-0 files will be mapped into a buffer with probability 1, and Rank-r files will not be mapped at all. Since unneeded files have been filtered out before mapping, the mapped files should survive in the buffer with probability 1.

EIRQ method mainly consists of four algorithms, with its working process being shown in Fig. 2. Since algorithms QueryGen and ResultDivide are easily understood, but algorithms MatrixConstruct and FileFilter are very much complex and they are analysed from the paper[3].In the FileFilter algorithm, for each file F_j, the cloud multiplies the rows that correspond to file keywords,elements.

- ✓ **QueryGen**
- ✓ ***MatrixConstruct***
- ✓ ***FileFilter***
- ✓ ***ResultDivide***

3.1 The EIRQ-Efficient Scheme:

EIQR-Efficient, two fundamental problems should be resolved: algorithms QueryGen and ResultDivide are easily understood,

Step 1. The user runs the QueryGen algorithm to send keywords and the rank of the query to the ADL. Since the ADL is assumed to be a trusted third party, this query will be sent without encryption.

Step 2. After aggregating enough user queries, the ADL runs the MatrixConstruct algorithm to send a mask matrix to the cloud. The mask matrix M is a d-row and r-column matrix,where d is the number of keywords in the dictionary, and r is the lowest query rank. Let M½i; j_ denote the element in the i-th row and the j-th column, and let l be the highest rank of queries that choose the i-th keyword Dic½i_ in the dictionary. M is constructed as follows: for the i-th row of M that corresponds to Dic½i_,M½i; 1_; . . .;M½i; r _ l_ are set to 1, and M½i; r_l þ 1_; . . .;M½i; r_ are set to 0, then each elements encrypted under the ADL's public key pk .For the rows that correspond to Rank-keywords, the ADL sets the first r _ l elements ,rather than random r _ l elements, to 1. The reason is to ensure that, given any Rank-file Fj, when we choose a random number k, the probability of all of the k-th elements of the rows that correspond Fj's keywords being 0 is l=r, which is determined by the highest rank of Fj's keywords.

Step 3. The cloud runs the FileFilter algorithm to return a buffer that contains a certain percentage of matched files to the ADL. Specifically ,the cloud multiplies the k-th elements of the rows that correspond to Fj's keywords to getherto form cj, where k ¼ j mod r. Then, it powers jFjj to cj to obtain ej, andmaps the c-epair into multiple entries of a buffer, as in the Ostrovsky scheme. Note that, with Step 2, we can make sure that, for a Rank-l file Fj, the probability of cj being 0 is l=r, and thus the probability of Fj being filtered out is l=r.

Step 4. The ADL runs the ResultDivide algorithm to distribute search results to each user. File contents are recovered as the File Recover algorithm in the Ostrovsky scheme. To allow the ADL to distribute files correctly, we require the cloud to attach keywords to the file content. Thus, the ADL can find out all of the files that match users' queries by executing keyword searches.

3.2 The EIRQ-Simple Scheme

The main differences lie in the MatrixConstruct and FileFilter algorithms (see Alg. 2). Intuitively, given queries that are classified into 0 _ r ranks, ADL sends r combined queries, denoted as Q0; . . .;Qr_1, to the cloud, each with a different rank. Specifically, for Qi, the ADL sets the j-th bit to an encryption of 1 if the j-th keyword Dic½j_ in the dictionary is chosen by at least one Rank-i query. The cloud then will generate r buffers, denoted as B0; . . . ;Br_1, each with a different file survival rate. Specifically, for Bi, the ADL adjusts the mapping time _i and the buffer size _i so that the survival rate of files in Bi is qi ¼ 1 _ i=r, where The main drawback of EIRQ-simple is that it returns redundant files when there are files satisfying more than one ranked query. For example, if Fi is of interest by Rank-0and Rank-1 queries, it will be returned twice (in Rank-0buffer and Rank-1 buffer, respectively), which wastes the network bandwidth. Therefore, the best case scenario is when there are no files of interest to different

ranked queries, and the worst case scenario is when queries of different ranks query the same files.

3.3 The EIRQ-Privacy Scheme

The working process of EIRQ-Privacy is similar to Fig. 2b.The main differences lie in the MatrixConstruct and FileFilter algorithms (see Alg. 3). Intuitively, EIRQ-Privacy adopts one buffer, with different mapping times for files of different ranks. Let _i denote the mapping times for a Rank I query, and let l be the highest rank of queries that choose the i-th keyword Dic½i_ in the dictionary. The mask matrix M is a d-row and m-column matrix, where d is the number of keywords in the dictionary, and m ¼ max _i. The Matrix Construct algorithm constructs M in the following way: for the i-th row of M that corresponds to Dic½i_, the ADL sets M½i; 1_; . . .;M½i; _l_ to 1, and M½i; _l þ 1_; . . . ;M½i;m_ to 0, and then encrypts each element under its public key. Note that for a row that corresponds to a Rank-lkeyword, the ADL sets the first_l elements, rather the random _l elements, to 1. The reason is to ensure that, given any Rank-l file, when we multiply the rows that correspond to file keywords together in a element-by-element way, the resulting row contains _l elements whose values are larger.

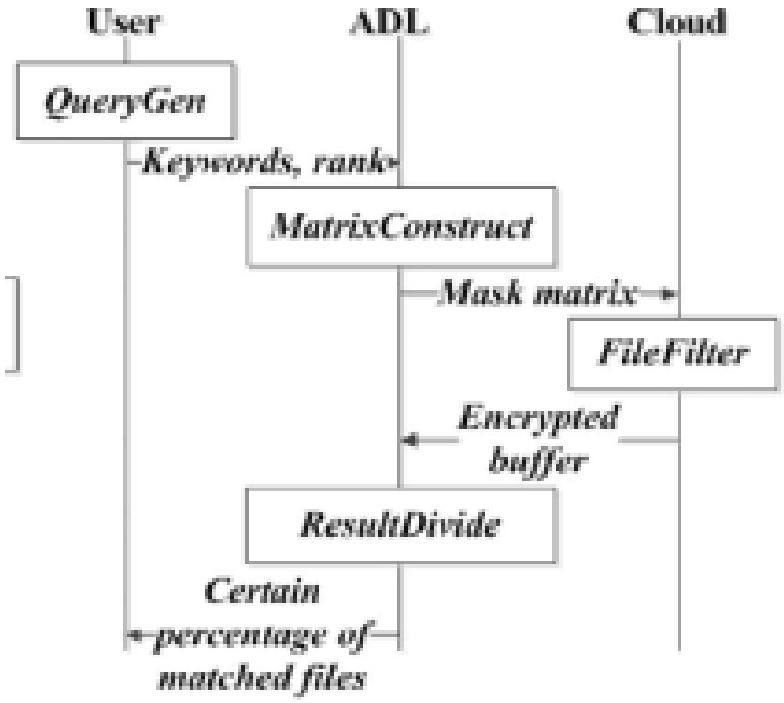

Fig 2: Working process of EIRQ scheme

V. RESULT

Since queries are classified into 0- 4 ranks, queries inRank-0, Rank-1, Rank-2, Rank-3, and Rank-4 should retrieve 100 percent, 75 percent, 50 percent, 25 percent,0 percent of matched files, respectively. However, in Fig. 3,the real failure rate in EIRQ-Simple and EIRQ-Privacy under the Ostrovsky parameter setting is much lower than i=r, and thus, the real file survival rate is higher than the desired value of 1 _ i=r (about 25 percent and 50 percent of files are redundantly returned to users); Only EIRQ Efficient, which filters a certain percentage of matched files before mapping them to a buffer ,provides differential query services .Under the Bloom filter parameter setting, we first obtain corresponding mapping times. Specifically, for file survival rate 100 percent, 75 percent, 50 percent, 25 percent, we have the optimal

mapping times 7, 2, 1, 0.4, respectively. Based on these values, the buffer size can be calculated with Eqs. (4), (5), and (6) for different schemes.

To verify the feasibility of our schemes, we deploy our program in Amazon EC2, to test the transfer-in (receiving query) and transfer-out (sending buffer) time at the cloud. The local machine has an Intel Core 2 Duo E8400 3.0 GHz CPU and 8 GB Linux RAM. We subscribe EC2 amzn-ami-2011.02.1.i386-ebs (ami-8c1fece5) AMI and a small type instance with the following specifications: 32-bit platform ,a single virtual core equivalent to 1 compute unit CPU, and1.7 GB RAM. The average bandwidth from EC2 to the local machine is 33.43 MB/s, and from the local machine to EC2is 42.98 MB/s.

First, we test the transfer-in time in the real cloud, which his mainly incurred by receiving queries from the ADL. Under both parameter settings, the query size for No Rank ,EIRQ-Simple, EIRQ-Privacy, and EIRQ-Efficient can be calculated with. Given d ¼ 100, r ¼ 4, and jwj ¼ 1 KB, the query size for No Rank, EIRQ-Simple, and EIRQ-Efficient is about 100 KB, 400 KB, and 400 KB, respective .

VI. CONCLSION

Cloud computing is used for sharing and retrieving information. In this paper we present different Techniques for searching over outsourced encrypted data. This study concludes that rank based retrieval is most efficient for searching on encrypted data because it is more secure, fast search access and does not leak information to un trusted authorities. However, while retrieving information from cloud environment it is necessary to get desired information with optimal communication and computation cost. In this paper, we have analyzed various algorithms which is used for efficient information retrieval in cloud environment. We have also shown the comparison of these algorithms which is useful for better understanding of these algorithms in terms of In EIRQ Scheme which is based on ADL to provide differential query services while protecting user privacy. By using this scheme user can retrieve different percentages of matched files by specifying queries of different ranks. By further reducing the communication cost incurred on the cloud, the EIRQ schemes make the private searching technique more applicable a cost-efficient cloud environment.

REFERENCES

1. P.Mell and T. Grance, ‘‘The NIST Definition of Cloud Computing (Draft),’’ in NIST Special Publication. Gaithersburg, MD, USA: National Institute of Standards and Technology, 2011.
2. R. Curtmola, J. Garay, S. Kamara, and R. Ostrovsky, ‘‘Searchable Symmetric Encryption: Improved Definitions and Efficient Constructions,’’ in Proc. ACM CCS, 2006, pp. 79-88.
3. R. Ostrovsky and W. Skeith, ‘‘Private Searching on Streaming Data,’’ in Proc. CRYPTO, 2005, pp. 233-240.
4. R. Ostrovsky and W. Skeith, ‘‘Private Searching on Streaming Data,’’ J. Cryptol., vol. 20, no. 4, pp. 397-430, Oct. 2007.
5. J. Bethencourt, D. Song, and B. Waters, ‘‘New Constructions and Practical Applications for Private Stream Searching,’’ in Proc IEEE SP, 2006, pp. 1-6.
6. M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. K. And Andy Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, “Above the clouds: A berkeley view of cloud computing,” Technical Report, University of Berkerley, 2009.

AUTHORS

Ms.M.SANDHYA received B.Tech in Computer Science and Engineering from the audisankara institute of technology affiliated to the Jawaharlal Nehru technological university Anantapur, in 2012, and pursing M. Tech in Computer Science and Engineering from visvodaya college of engineering affiliated to the Jawaharlal Nehru technological university Anantapur in 2015, respectively.

Mr.D.ANIL has received Degree from JNTU Hyderabad and M.E degree in computer science and engineering at anna university Chennai in 2008. He is dedicated to teaching field from the last 6 years. He has Guided 8P.G and 13 U.G students. His research areas included cloud computing, design of algorithms and web environment applications. At present he is working as Associate Professor in visvodaya engineering college, Kavali, and Andhra Pradesh, India.

