ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

Query Services in Cost Efficient Cloud Using Query Analysis

VanthanaPriya.J[1], ArunKumar.B[2]

PG Scholar, Department of CSE, Karpagam University, Coimbatore, Tamil nadu, India[1]

Assistant Professor, Department of CSE, Karpagam University, Coimbatore, Tamil nadu, India[2]

ABSTRACT—Cloud Architecture makes trends in the IT Industries in order to provide a cost efficient environment. This makes the environment so faster data transaction and secured also. This project is entirely based on cost efficient cloud environment. This majorly happens in IT industries, for creating queries the developer takes too long time. This is because still SQL works on dos prompt mode. And there is no IDE – Integrated development environment. SQL Server response for one user at one time while generating code. The basic fundamental issues are Privacy – Efficiency - Analysis - Cost reduction. In this paper a scheme, termed as data retrieval for ranked query (DRQ), and based on an aggregation and distribution layer (ADL), to reduce querying overhead incurred on the cloud. So that in DRQ, queries are classified into multiple ranks, for that data acquisition will done in a higher ranked queries. Here the user can retrieve files on demand by choosing queries of dissimilar ranks. This feature is useful when there are a large number of matched files, but the user only needs a small subgroup of them. This scheme describes a SQL-compiler and a query generator which can be access through query aggregation methods. This compiler can translate the query language of SQL to the programming language. Two options are provided for the programmer using the SQL-compiler. SQL statements can both be embedded in SQL online code generator and compiled together alone with the compiled query at run time when the query is evaluated in the cloud server. The implementation of this application can be made through 3 tier architecture design. It includes data layer, Business layer and Application layer in the cloud architecture.

KEY WORDS— cloud computing, cost efficiency, query services.

I. INTRODUCTION

Private searching was already discussed in ostrovsky scheme [1],[2].An untrusted server is provided by the client with an encrypted query. The server uses the query on a stream of documents and returns the matching documents to the client while learning nothing about the nature of the query. The server will then run a search algorithm on a stream of files while keeping an encrypted buffer storing information about files for which there is a keyword match. The encrypted buffer will then be returned to the client to enable the client to reconstruct the files that have matched his query keywords. User privacy is classified into search privacy and access privacy [3]. Search privacy means that cloud does not know about what the user is searching for, and access privacy means that cloud does not know about which files are returned to the user. When files are stored in the clear forms, a simple solution to user privacy is for the user to request all the files from the cloud, in this way the cloud cannot know which file the user is interested in. Because of the necessary privacy, the communication cost is high [4]. In single-database PIR keyword search, the size of the stream must be the independent of the program size, as the stream is supposed to be an arbitrarily large source of data and we do need to know the size of the stream when compiling the obfuscated query. In previous non-trivial PIR protocol when creating the PIR query, the user of this protocol must know the upper bound on the database size. If a single keyword is searched in a database, single result is returned. But in single-database PIR keyword search, if one wanted to query an "OR" of several keywords, this would require creating several PIR queries, and then sending each to the database. We however show how to intrinsically extend the types of queries that can be performed, withoutloss of efficiency or with multiple queries. New Private stream searching achieves the optimal communication from the server to the client and server storage overhead in returning the content of the matching documents, given any fixed probability of successfully retrieving all matching documents. A pseudo random function is shared by the client and the

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

server, will determine the probability whether the document is copied into a given location, where the function takes as input the document number and buffer location [5].

Private searching is applicable in cloud environment by using Cooperative private searching protocol (COPS) [6].This protocol provides privacy protection with much lower overhead. In addition it reduces the computation and communication cost. This protocol allows multiple users to combine their queries to reduce the querying cost. Aggregation and distribution layer (ADL) is a proxy server that is introduced between user and the cloud. The user will send the queries to the ADL, which will combine the query on user's behalf. The cloud need to execute query only once to return matching files of all users queries to the ADL. Under ADL, the files interested by many users need to be returned only once. The COPS protocol will incur some processing delay for aggregating queries. The time-out mechanism is used to control the degree of aggregation. Security protocols such as SSL used to protect user privacy during transferring of information.

Searchable symmetric encryption (SSE) [3] allows a party to outsource the storage of data to another party in a private manner, while maintaining the capacity to selectively search over it. The first construction is the most efficient non-adaptive SSE scheme to date in terms of computation on the server, and acquires a minimal cost for the user. The second construction achieves adaptive security, which was not previously achieved by any constant-round solution. The problem of SSE is extended to the multi-user setting, where a client wishes to allow an authorized group of users to search through its document collection. In prior work, SSEonly considered the setting where only the owner of the data is capable of submitting search queries. By considering the natural extension where an arbitrary group of parties other than the owner can submit search queries.

II. RELATED WORK

Our work aims to provide differential query services while protecting user privacy from the cloud. Research of query services is already found in areas of private searching [1]-[11]. Unlike private-key encryption [3], [8] prevents one from searching over encrypted data, clients also lose the ability to selectively retrieve segments of their data, private searching performs keyword-based searches on unencrypted data. Private searching was first proposed in [1], [2], which allows the server to filter streaming data without conceding user privacy. Their solution requires the server to return a buffer of size O(f log(f)) when f files match a users query. Each file is related with a survival rate, which denotes the probability of this file being successfully recovered by the user. Based on the paillier cryptosystem [13], the mismatched query of a file will not survive in the buffer, but the matched files enjoy the file survival rate.

Ostrovsky scheme was introduced in [3],[4], which relies on a paillier cryptosystem[13]. Plain text and Public key is used. The paillier cryptosystem allows the routine of certain operations such as multiplication and exponentiation, on ciphertext directly. The ciphertext is converted into plaintext. The user obtains the plaintext that processes addition and multiplication operations. This scheme consists of three algorithms. Two assumptions are made in this scheme: first, a dictionary consists of universal keywords is supposed to be publicly available. Second, the users are supposed to have the ability to estimate the number of files that match their queries.

The three algorithms are:

GenerateQuery: The user runs this algorithm to send an encrypted query to the cloud. The query is a bit string encrypted under the users public key. If the keyword in the dictionary is chosen, then each bit is an encryption of 1; otherwise, it is an encryption of 0.

PrivateSearch: The cloud runs this algorithm to return the encrypted buffer to the user. The cloud processes the encrypted query on every file to generate an encrypted c-e pair, and maps it to multiple entries of an encrypted buffer.

FileRecover: The user runs this algorithm to recover files. The user decrypts the buffer, entry by entry, to obtain the plaintext c-e pairs. The buffer size depends only on the matched files.

Ref [10] presented the efficient decoding private search mechanism that allows file recover that collide in a buffer position. Ref [11] proposed a recursive mechanism which requires a buffer of size O(f) when f files match a users query.

Ref [12] proposed two new communication optimal construction, Read-Solomon code allows for zero error, and irregular LDPC codes allows for lower computation cost at the server. Private searching on streaming data only support searching for OR of keywords or AND of two sets of keywords. Extend the private queries to support streaming of an OR of a set of both single and conjunctive keywords. Ref. [15] queries are extended to support disjunctive normal

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

forms (DNF) of keywords. The drawback of this scheme is that both the communication and computation cost grows linearly with the number of executing queries. The querying cost will be extensive when these schemes are applied to large-scale cloud environment. Private searching techniques are already applicable to a cloud environment. Ref. [6]. The cloud need to execute query only once to return matching files of all users queries to the ADL. It may cause a waste of bandwidth when only a small percentage of files are of interest. The time-out mechanism is used to control the degree of aggregation. Security protocols such as SSL used to protect user privacy during transferring of information. To overcome this problem a differential query services is introduced in [14].This scheme is used to reduce the querying overhead incurred on the cloud. The user can retrieve file based on higher ranks.

III. SCHEME DESCRIPTION

The basic idea of DRQ Efficient is the construction of mask matrix. Before mapping them to a buffer, the cloud is able to filter out a certain percentage of matched files.

```
DRQ-Efficient Scheme:
Maskmatrix M
    Keywords in Dic d
    Keyword in each query k
    Lowest rank r
    Highest rank of queries choosing Dic[a] is l
    Buffer size B
    Mapping time R
    Rank a
    Encryption under public key $E_{pk}$
    $a_{th}$ row and $g_{th}$ column M[a,g]
    Buffer size in ADL is B(a)
    Mapping times is R(a)
    File survival rate=1-(a/r)

    1)  ConstructMatrix
    2)  a←1 to d
    3)  g←1 to r
    4)  if (g < = r-l)
    5)      M[a,g]= $E_{pk}$ (1)
    6)  else
    7)      M[a,g]=$E_{pk}$(0)
    8)  Adjust R and B so file survival rate is 1
    9)  FileFilter
    10) File $F_j$ stored in the cloud
    11) a←1 to d
    12) k=g mod r
    13) $c_j=\pi_{Dic[a] \in F_j}$ M [a,k]
    14) map ($c_{g,} e_g$) R times to a buffer of size B
```

Two basic problems should be resolved: First, we should find the percentage of retrieved matched files and query rank. Queries are classified into 0~r ranks. Highest rank is occupied by Rank-0 queries and lowest rank is by Rank-r queries. Rank-a queries can retrieve (1-a/r) percent of matched files. 100% of matched files are by rank-0 queries and rank-r query cannot retrieve any files. Second, we should determine whether the matched files should be returned or not. The probability of the file can be determined by highest rank of queries matching the file. Specifically keywords should be ranked first by choosing highest rank of queries and then rank each file by highest rank of keywords. Therefore, with the probability 1, Rank-0 files are mapped into a buffer, and Rank-r files will not get mapped. Before mapping the unnecessary files are filtered out, the mapped files will survive in buffer with probability 1.

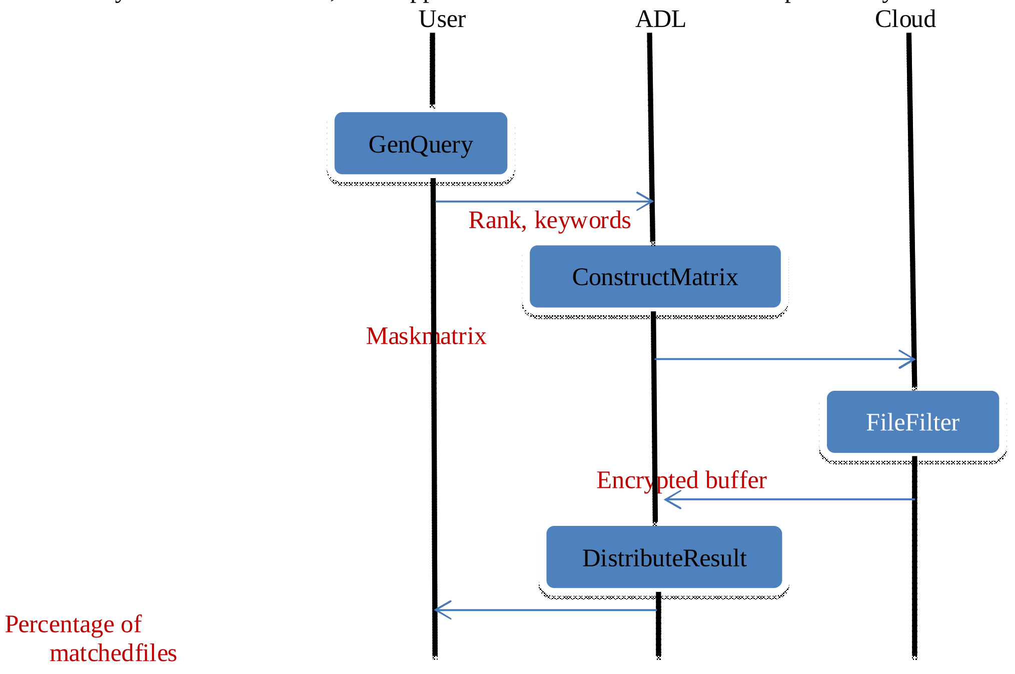

Fig 1. The DRQ-Efficient Scheme

DRQ-Efficient consists of four algorithms, the working process is shown in Fig 1.

Step 1: The user runs the GenQuery algorithm to send rank of the query and keywords to the ADL. The query can be sent without encryption, with the help of ADL.

Step 2: After combining the user queries, the ADL runs the ConstructMatrix algorithm to send maskmatrix to the cloud. The maskmatrix M consists of d-row and r-column matrix, where d denotes the number of keywords in the dictionary, and r denotes the lowest query rank. Let M[a,g] denote the element in the a-th row and j-th column and l be the highest rank of queries that choose a-th keyword in the Dic[a] in the dictionary. Maskmatrix M is constructed as follows: for a-th row of M that corresponds to Dic[a], M[a,l],…,M[a,r-l] are set to 1, and M[a,r-l+1],….,M[a,r] are set to 0. Rather than choosing the random r-l elements, the ADL sets the first r-l element to 1. The probability of k-th element that corresponds F_j keywords being 0 is l/r.

Step 3: The cloud runs the FileFilter algorithm to return the buffer. The buffer also contains the matched files to the ADL. Corresponds to $F_j$keyword the cloud multiplies the k-th element to form c_j, where k= g mod r. In ostrovsky scheme, c-e pairs into the multiple entries of the buffer. From the step, Rank-l file $F_{j,}$ the probability of $c_j$being 0 is l/r, thus the probability of $F_j$being filtered out is l/r.

Step 4: The ADL runs the DistributeResult algorithm to distribute the search results to each user. The cloud attaches the keywords to the file contents, so the ADL is able to distribute files correctly. By executing user queries, the ADL can find all matched of the users queries.

IV. CONCLUSION AND FUTURE WORK

Our proposal of DRQ scheme is based on ADL to provide differential query services while protecting search privacy and access privacy. By specifying queries of different rank, a user can retrieve different percentage of matched files. The communication cost is reduced in cloud with the help of DRQ scheme. This makes the private searching technique will be more applicable in cost efficient cloud environment. In the future work we will try to propose an effective ranking mechanism for DRQ scheme and SQL compiler for query aggregation method.

REFERENCES

[1] R. Ostrovsky and W. Skeith, "Private searching on streaming data," in Proc. of CRYPTO, 2005.
[2] ——, "Private searching on streaming data," Journal of Cryptology, 2007.
[3] R. Curtmola, J. Garay, S. Kamara, and R. Ostrovsky, "Searchablesymmetric encryption: improved definitions and efficient constructions," in Proc. of ACM CCS, 2006.
[4] E. Kushilevitz and R. Ostrovsky. Replication is not needed: Single database, computationally-private information retrieval. In Proc. of the 38th Annu. IEEE Symp. OnFoundations of Computer Science, pages 364–373, 1997.
[5] J. Bethencourt, D. Song, and B. Waters, "New constructions andpractical applications for private stream searching," in Proc.Of IEEE S&P, 2006.
[6] Q. Liu, C. Tan, J. Wu, and G. Wang, "Cooperative private searching in clouds," Journal of Parallel and Distributed Computing, 2012.
[7] P. Mell and T. Grance, "The nist definition of cloud computing(draft)," NIST Special Publication, 2011.
[8] B. Hore, E.-C.Chang, M. H. Diallo, and S. Mehrotra, "Indexingencrypted documents for supporting efficient keyword search," in Secure Data Management, 2012.
[9] ——, "New techniques for private stream searching," ACM Transactions on Information and System Security, 2009.
[10] G. Danezis and C. Diaz, "Improving the decoding efficiency ofprivate search," in IACR Eprint archive number 024, 2006.
[11] ——, "Space-efficient private search with applications to rateless codes," Financial Cryptography and Data Security, 2007.
[12] M. Finiasz and K. Ramchandran, "Private stream search at thesame communication cost as a regular search: Role of ldpc codes,"in Proc. of IEEE ISIT, 2012.
[13] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in Proc. of EUROCRYPT, 1999.
[14] Q. Liu, C. C. Tan, J. Wu, and G. Wang, "Efficient informationretrieval for ranked queries in cost-effective cloud environments," in Proc. of IEEE INFOCOM, 2012.
[15] X. Yi and E. Bertino, "Private searching for single and conjunctive keywords on streaming data," in Proc. of ACM Workshop on Privacyin the Electronic Society, 2011.
[16] G. Wang, Q. Liu, J. Wu, and M. Guo, "Hierarchical attribute-based encryption and scalable user revocation for sharing data in cloud servers," Computers & Security, 2011.
[17] M. Mitzenmacher, "Compressed bloom filters," IEEE/ACM Transactions on Networking, 2002.
[18] D. Guo, J. Wu, H. Chen, and X. Luo, "Theory and network applications of dynamic bloom filters," in Proc. of IEEE INFOCOM,2006.
[19] A. Berl, E. Gelenbe, M. Di Girolamo, G. Giuliani, H. De Meer,M. Q. Dang, and K. Pentikousis, "Energy-efficient cloud computing," The Computer Journal, 2010.
[20] E. Gelenbe, R. Lent, and M. Douratsos, "Choosing a local orremote cloud," in Proc. of IEEE NCCA, 2012.