International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 4 Issue 03, March-2015

Achieving Privacy & Efficiency in Cost-Efficient Cloud Environment

G D Ravi Teja[1]
Post Graduate Student,
Department of Computer Science & Engineering,
PVKK Institute of Technology,
Anantapuramu, Andhra Pradesh, India

Ishaq Shareef C[2]
Assistant Professor,
Department of Computer Science & Engineering,
PVKK Institute of Technology,
Anantapuramu, Andhra Pradesh, India

***Abstract*— Cloud computing is a recently evolved computing terminology & it's expected to reshape information technology trends in future. While retrieving information from the cloud a user can tolerate a certain percentage of delay. In such an environment we address two fundamental issues: One is *Privacy* & the other is *Efficiency*. Before proceeding, let's firs review the concept proposed by Ostrovsky. Ostrovsky scheme is a private keyword-based file retrieval scheme that allows users to retrieve files without leaking their information to the untrusted server. Drawback of this scheme is, it causes heavy querying overhead on the cloud & thus violates our goal of Efficiency. To overcome the defects in Ostrovsk scheme, we present a scheme named Efficient Information retrieval for Ranked Query (EIRQ). Here we use a proxy server called aggregation and distribution layer (ADL) in order to reduce querying overhead on the cloud. EIRQ queries are categorized into multiple ranks, where a higher ranked query retrieves a higher percentage of matched files and lower ranked query retrieves a lower percentage of matched files. Based on the demand a user can retrieve files by choosing queries of different ranks. This feature is helpful when there are a huge number of matched files, but the user needs only a subset of them.**

Keywords*— Cloud computing, privacy, cost efficiency, ranking scheme

I. Introduction

Overwhelming merits of cloud computing such as cost effectiveness, scalability and flexibility made most organizations choose to outsource their data for sharing in the cloud. An organization subscribes to the cloud services and authorizes its staff to share files in the cloud. In such an environment, how to protect privacy of user in the cloud, which is outside the security boundary of an organization.

Private searching was proposed by Ostrovsky et al which allows a user to retrieve files of interest from an untrusted server without leaking any information. Ostrovsky scheme has high computational cost and requires the cloud to process the query on every file in a collection. Else, the cloud learns that certain files unprocessed, are of no interest to the user.

To make private searching applicable in a cloud environment we use 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 set up inside an organization has two main functionalities: One is aggregating user queries and the other is distributing search results. Here we use innovative concept called differential query services to COPS, where users are allowed to personally decide number of matched files to be returned.

We propose a system named Efficient Information retrieval for Ranked Query (EIRQ), in which each user will be allowed to choose the rank of his query in order to determine the percentage of matched files to be returned.

II. RELATED WORK

Our work intents to provide differential query services while protecting user privacy from the cloud. A user performs keyword-based searches on unencrypted data in private searching. When private searching was initially proposed it allowed to filter streaming data without compromising user privacy. Ostrovsky solution requires the server to return a buffer of size O(f log(f)) when f files match a user's query. Each file returned contains a survival rate, which denotes the probability of file being successfully recovered by the user. Based upon Paillier cryptosystem, files that do not match a query will not survive in the buffer, but the matched files enjoy a higher survival rate. The major drawback of existing private searching schemes is that both the computation and communication costs grow proportionally with the number of users executing queries. To overcome this problem, we introduce the concept of differential query services through Aggregation and Distribution Layer.

III. ARCHITECTURE

A. System Model

The system primarily consists of three entities: aggregation and distribution layer (ADL), multiple users, and the cloud, as shown in Fig. 1. We only use a single ADL for our analysis, but multiple ADLs can be deployed as needed. The ADL implemented in an organization authorizes its staff to share data in the cloud. User queries are aggregated in the ADL & combined queries are sent to the cloud. The cloud then processes the combined query and returns a buffer which contains all of matched files to the ADL, which will distribute the retrieved files to each user accordingly.

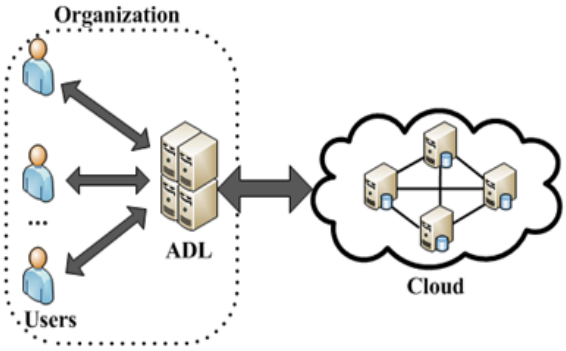

Fig. 1. System model

B. Design Goals

User privacy can be classified into search privacy and access privacy. In our trial, user queries are classified into multiple ranks, and thus a new type of user privacy, called rank privacy, also needed. Rank privacy requires to hide the rank of each user query sent to the cloud through ADL, i.e., cloud provides differential query services unknowingly which level of service is chosen by the user. Rank privacy can be further classified into basic level and high level, where the basic level will hide the rank of each query, and high level will further hide the number of ranks from the cloud. Our design goal can be subdivided as below:

- Cost efficiency. Users can retrieve only matched files on demand, which reduces the communication costs provoked on the cloud.
- User privacy. Cloud does not know anything about the user's search keywords, returned queries, and at least the rank.

C. Overview of Ostrovsky Scheme

Here we briefly introduce Ostrovsky scheme, which depends on a public key cryptosystem, the *Paillier cryptosystem*. The Paillier cryptosystem allows a kind of operations, such as *multiplication* and *exponentiation*, on cipher-text directly. For given cipher text, user can obtain the corresponding plaintext by processing addition and multiplication operations.

The Ostrovsky scheme consists of three algorithms, the working process is shown in Fig. 2-(a). Two assumptions are used in this scheme: first, a dictionary consisting of universal keywords is assumed to be available publicly; second, users are assumed to have the capability to estimate the number of files that match their queries.

Step 1. The user runs the *GenerateQuery* command to send an encrypted query to the cloud. The query is a bit string encrypted with the user's public key, where each bit of string is an encryption of 1, if the keyword from dictionary is chosen; or else, it is an encryption of 0.

Step 2. The cloud runs the *PrivateSearch* command to return an encrypted buffer to the user. In general, the cloud processes the encrypted query on every file in the collection to generate an encrypted pair, and maps it to multiple entries of an encrypted buffer.

Step 3. User runs the *FileRecover* command to retrieve files. The user decrypts the buffer, entry by entry, to get the plaintext. For the entries in survival state, file content can be retrieved by dividing the plaintext value by value.

The security of Ostrovsky scheme is derived from the *semantic security* of Paillier cryptosystem. The key technique of their scheme is, the files not matching user's query are encrypted 0s, which will have no impact on the matched files. Thus, the buffer size depends only on the number of matched files, which will be much smaller than the total number of files stored in the cloud.

IV. DESCRIPTION OF SCHEMES

Here let us discuss the original EIRQ scheme and its two extensions. To distinguish the three schemes, we name the original EIRQ scheme as EIRQ-Efficient, and the extensions as EIRQ-Simple, and EIRQ-Privacy.

A. The EIRQ-Efficient Scheme

Before highlighting EIRQ-Efficient, two fundamental problems needs to be resolved:

Firstly, we should determine the relationship between query rank and the percentage of matched files to be returned. Suppose that queries are divided into $0 \sim r$ ranks. Rank 0 queries should have the highest rank and Rank r queries should have the lowest rank. We determine this relationship by allowing Rank i queries to return $(1 - i/r)$ percent of matched files. Therefore, Rank 0 queries can return 100% of matched files, whereas Rank r queries cannot.

B. The EIRQ-Simple Scheme

The working flow of EIRQ-Simple is similar to Fig. 2-(b). Given queries are classified into $0 \sim r$ ranks, the ADL sends r combined queries, denoted as Q0, ..., Qr−1, to the cloud, each contained with a different rank. Specifically, for query Qi, the ADL sets the j-th bit to an encryption of 1. If the j-th keyword Dic[j] in the dictionary is chosen by at least one Rank i query then the cloud will generate r buffers, denoted as B0,...,Br−1, each with a dissimilar file survival rate. Specifically, for Bi, ADL adjusts the mapping time γi and the buffer size βi so that the survival rate of files in buffer Bi is $qi = 1 - i/r$, where $0 \le i \le r - 1$.

The main drawback of EIRQ-Simple is that it retrieves duplicate files when there are files matching more than one ranked query.

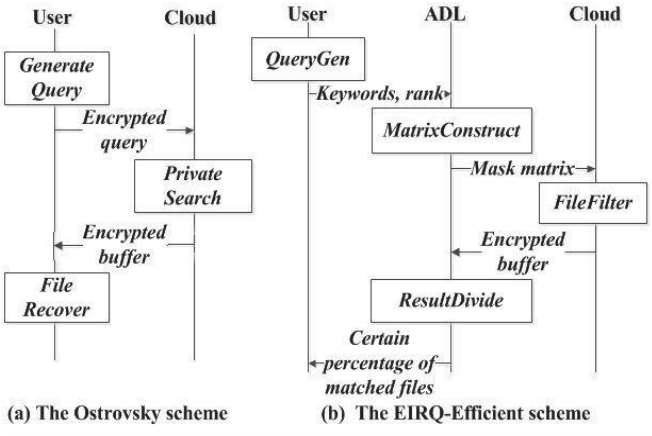

Fig. 2. Working process

C. The EIRQ-Privacy Scheme

The work flow of EIRQ-Privacy is similar to Fig. 2-(b). The difference between two schemes lies in maintaining a buffer. EIRQ-Privacy maintains one buffer, with different mapping times for files of different ranks whereas EIRQ-Simple won't.

V. SECURITY & PERFORMANCE ANALYSIS

By our work below, we will prove that EIRQ schemes can provide search, access, and rank privacy to the user in cloud along with efficiency.

Search privacy. In all three schemes, combined query sent to the cloud is encrypted in ADL by using public key with the Paillier cryptosystem. Now the combined query is an encrypted matrix consisting of 0s and 1s. As Paillier cryptosystem is semantically secure, and the ciphered-text of every 1 or 0 is different from other 1s or 0s. We therefore, consider cloud cannot find what each user is searching for.

Access privacy. In all three schemes, the cloud processes encrypted query on each file in a collection, and then maps the processing result to the buffer, which is encrypted with ADL's public key. The cloud applies this procedure for all files in same way. Therefore, the cloud does not know which files are actually retrieved from the encrypted buffer.

Rank privacy. In EIRQ-Simple, messages from ADL to cloud are r encrypted queries, with buffer size, and the mapping times, where r is the data, which we leak to the cloud. Given r, the cloud only learns the number of query ranks without knowing number of users in each rank, nor which users are in which ranks. Thus, EIRQ-Simple can protect the basic level of rank privacy for a user. In EIRQ-Privacy, messages from ADL to the cloud is a mask matrix with d-rows and m-columns, where d is the number of keywords in the dictionary, and m = maxγi the maximal value of mapping times.

Performance comparison between No Rank and the three EIRQ schemes can be done by using different parameter settings. In No Rank, ADL only combines the user queries, but do not provide differential query services. Suppose that queries are categorized into 0 ~ r ranks, t files are stored in the cloud whose keywords create a dictionary of size d, and fi files matching Rank i queries but mismatching higher ranked queries. Moreover, in No Rank and EIRQ-Efficient schemes, the threshold file survival rate p0 is set to α; in EIRQ-Simple and EIRQ-Privacy, pi is set to i/r+α.

VI. EVALUATION

This section is intend for comparing three EIRQ schemes from the following aspects: file survival rate and computation/communication cost observed on the cloud.

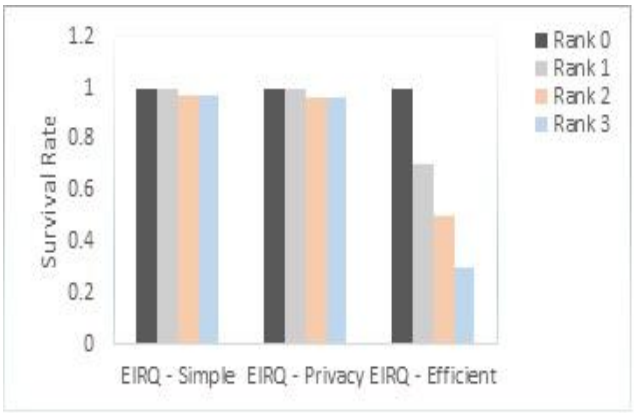

Fig. 3. File survival rate under Ostrovsky setting

A. File Survival Rate

As an example let us consider queries which are having 0 ~ 4 ranks, queries in Rank 0, Rank 1, Rank 2, Rank 3, and Rank 4 should retrieve 100%,75%,50%,25%,0% of matched files, respectively. The real failure rate in EIRQ-Simple and EIRQ-Privacy under the Ostrovsky parameter setting is much lower than i/r, and therefore, the real file survival rate is higher than the desired value of 1 − i/r); Only EIRQ-Efficient, that filters a certain percentage of matched files before mapping to a buffer, provides differential query services.

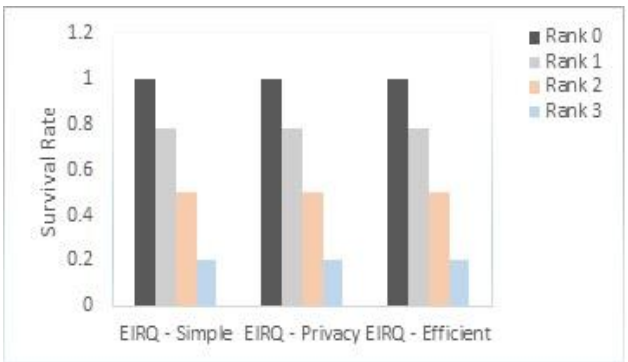

Fig. 4. File survival rate under Bloom filter setting

Under Bloom filter parameter setting, we first obtain comparable mapping times. Specifically, for file survival rate 100%,75%,50%,25%, we have the optimal mapping times of 7,2,1,0.4, respectively.

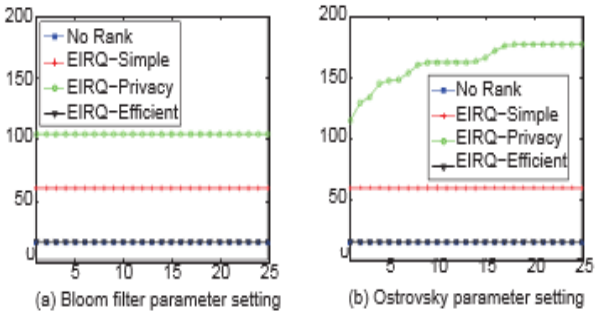

Fig. 5. Comparison of computational cost at the cloud. The x-axis denotes the number of queries in each rank, and the y-axis denotes the computation time (s).

B. *Computational & Communication Cost*

Computational cost can be determined by the number of exponential equations performed by the cloud, which is almost same under the Bloom filter and the Ostrovsky parameter settings. In both the settings, EIRQ-Privacy consumes the most computation cost, and like No Rank, EIRQ-Efficient consumes the least computational cost.

Communication cost mainly depends upon the buffer size generated by the cloud, which can be calculated in different ways under different parameter settings. The buffer size depends upon the number of files that match the user queries, which is distinct when users have different common interests.

VII. CONCLUSION

In this paper, we proposed how user privacy can be achieved without impacting efficiency while retrieving information from the cost efficient clouds by using ADL. Our three EIRQ schemes, helps the user to retrieve different percentages of matched files by specifying ranks of queries before sent to the ADL. To aggregate sufficient users' queries, the organization may need the ADL to wait for a certain time before running our schemes, which may cause a certain querying delay. For our future work, we will try to design flexible mechanisms for achieving user privacy & efficiency with minimal querying delay.

REFERENCES

1. P. Mell and T. Grance, "The nist definition of cloud computing (draft)," *NIST Special Publication*, 2011.
2. R. Curtmola, J. Garay, S. Kamara, and R. Ostrovsky, "Searchable symmetric encryption: improved definitions and efficient con-structions," in *Proc. of ACM CCS*, 2006.
3. R. Ostrovsky and W. Skeith, "Private searching on streaming data," in *Proc. of CRYPTO*, 2005.
4. ——, "Private searching on streaming data," *Journal of Cryptology*, 2007.
5. J. Bethencourt, D. Song, and B. Waters, "New constructions and practical applications for private stream searching," in *Proc. of IEEE S&P*, 2006.
6. ——, "New techniques for private stream searching," *ACM Trans-actions on Information and System Security*, 2009.
7. Q. Liu, C. Tan, J. Wu, and G. Wang, "Cooperative private search-ing in clouds," *Journal of Parallel and Distributed Computing*, 2012.
8. X. Yi and E. Bertino, "Private searching for single and conjunctive keywords on streaming data," in *Proc. of ACM Workshop on Privacy in the Electronic Society*, 2011.
9. B. Hore, E.-C. Chang, M. H. Diallo, and S. Mehrotra, "Indexing encrypted documents for supporting efficient keyword search," in *Secure Data Management*, 2012.
10. Q. Liu, C. C. Tan, J. Wu, and G. Wang, "Efficient information retrieval for ranked queries in cost-effective cloud environments," in *Proc. of IEEE INFOCOM*, 2012.
11. S. Yu, C. Wang, K. Ren, and W. Lou, "Achieving secure, scalable, and fine-grained data access control in cloud computing," in *Proc. of IEEE INFOCOM*, 2010.
12. 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.
13. M. Mitzenmacher, "Compressed bloom filters," *IEEE/ACM Trans-actions on Networking*, 2002.
14. D. Guo, J. Wu, H. Chen, and X. Luo, "Theory and network ap-plications of dynamic bloom filters," in *Proc. of IEEE INFOCOM*, 2006.
15. A. Berl, E. Gelenbe, M. Di Girolamo, G. Giuliani, H. De Meer, M. Q. Dang, and K. Pentikousis, "Energy-efficient cloud comput-ing," *The Computer Journal*, 2010.
16. E. Gelenbe, R. Lent, and M. Douratsos, "Choosing a local or remote cloud," in *Proc. of IEEE NCCA*, 2012.