A COMPARATIVE STUDY OF SECURE SEARCH PROTOCOLS IN PAY-AS-YOU-GO CLOUDS

V. Anand[1], Ahmed Abdul Moiz Qyser[2]

[1]*Muffakham Jah College of Engineering and Technology, Hyderabad, India*
[2]*Muffakham Jah College of Engineering and Technology, Hyderabad, India*

Abstract

Cloud computing has emerged as a major driver in reducing the information technology costs incurred by organizations. In a cost-sensitive environment, an organization is willing tolerate a certain threshold of delay while retrieving information from the cloud. In this paper, we focus on the fundamental issues of cost efficiency and privacy. We review three keyword-based secure search protocols, namely Ostrovsky protocol, COPS protocol and EIRQ protocol. Each protocol offers an improvement over the basic keyword search used by today's cloud users. In Ostrovsky, encryption offers privacy. In COPS, aggregation results in a limited amount of cost efficiency. In EIRQ, users can retrieve the desired percentage of files by assigning ranks to queries. This feature is useful if the user is only interested in a subset of all the matched files. We present a comparison of the performance of these protocols and conclude by listing the future work that could be carried out in this area.

Keywords: *differential query services, cloud computing, cost efficiency, pay-as-you-go.*

----------***----------

1. INTRODUCTION

NIST defined Cloud Computing as "a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction" [1]. Organizations are storing their critical data on the cloud in an attempt to lower their capital expenditure outlays. A typical organization storing documents on the Infrastructure-as-a-Service (IaaS) cloud allows its users to access the documents from the cloud. Each document is described by a set of keywords. The users would query the cloud with certain keywords. The system model is depicted in Fig. 1.

In contrast to a normal cloud, a *pay-as-you-go cloud* allows an organization to pay the cloud service provider for the CPU and bandwidth consumed. It is similar to a metered utility service.

In this model, each user submits queries to the cloud and gets a response from the cloud. For example, a user query could be of the form "*Sales, New York, 2013*", where keywords are separated by commas. If the number of user queries is *n*, the number of round trips to the cloud is equal to *n*. This leads to excessive CPU and bandwidth costs for the cloud customers.

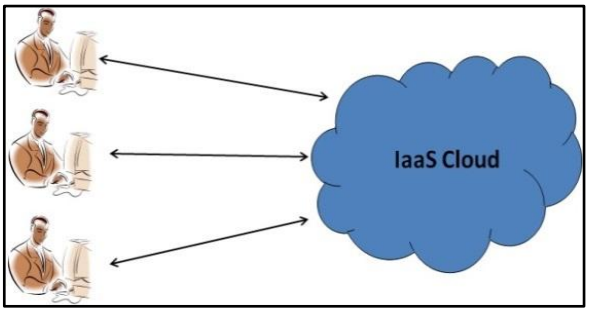

Fig 1 System Model of Cloud-based Keyword Search

The objectives of an organization are:

1. Increasing the privacy of the users by masking the search keywords in such a way that the cloud does not know the search patterns of individual users.
2. Decreasing the cost of CPU consumption on the cloud.
3. Decreasing the cost of network bandwidth usage.

In this paper, we provide a comparative study of several secure search protocols and identify a few areas that could be improved in future.

2. SECURITY MODEL

All the secure search protocols presented in this paper use an encryption scheme known as Paillier cryptosystem [2]. It is a public key cryptosystem with wide applications in cloud computing, electronic voting and other areas.

2.1 Paillier Cryptosystem

Consider two large primes *p* and *q* chosen in compliance with the following property:

$$gcd(pq, (p-1)(q-1)) = 1 \quad (1)$$

n is set as the product of *p* and *q*. If *g* and *r* are two random integers, then the plaintext *m* is converted to cipher text *c* using the following equations:

$$\lambda = lcm(p-1, q-1) \quad (2)$$

If *L* is a function of the form $L(u) = (u-1) / n$

$$\mu = (L (g^{\lambda} \bmod n^2))^{-1} \bmod n \quad (3)$$

$$\text{Cipher text } c = g^m . r^n \bmod n^2 \quad (4)$$

$$\text{Plaintext } m = L (c^{\lambda} \bmod n^2). \mu \bmod n \quad (5)$$

Let $E_{pk}(m)$ denote the encryption of a plaintext message *m* with public key *pk*. Paillier cryptosystem exhibits the following homomorphic properties:

$$Epk(a) \cdot Epk(b) = Epk(a + b) \quad (6)$$

$$Epk(a)^b = Epk(a \cdot b) \quad (7)$$

Unlike RSA cryptosystem, Paillier cryptosystem results in a non-zero cipher text for a plaintext message of value 0. This feature facilitates masking the absence of certain keywords in the user queries.

3. DATA ON THE CLOUD

Files belonging to an organization are stored on the cloud. A file could be of any format such as .doc, .pdf, .xml, .html and so on. The format does not have any impact on the processing done by the protocols presented in this paper.

Each file has a few keywords associated with it. These keywords are added to a dictionary. The dictionary is publicly available to both the organization and the cloud. Each time a new file is stored on the cloud, the dictionary is updated by the organization. Similarly, whenever a file is deleted from the cloud, the dictionary is updated.

4. SECURE SEARCH PROTOCOLS

We review three keyword-based secure search protocols, namely Ostrovsky, COPS and EIRQ.

4.1 Ostrovsky Protocol

Private searching was proposed by Rafail Ostrovsky in [3]. Ostrovsky protocol made two primary contributions. First, the size of data has no impact on the size of the program. Secondly, both the matching and non-matching files are processed the same way. This feature makes it difficult for the cloud to determine if the search condition is satisfied without breaking the encryption. It must be noted that the cloud is unaware of the private key of the organization.

Ostrovsky also introduced the notion of *file survival rate*, which denotes that the probability that a file would be successfully recovered by the user from the response buffer.

The three algorithms of Ostrovsky protocol work as shown in Fig. 2.

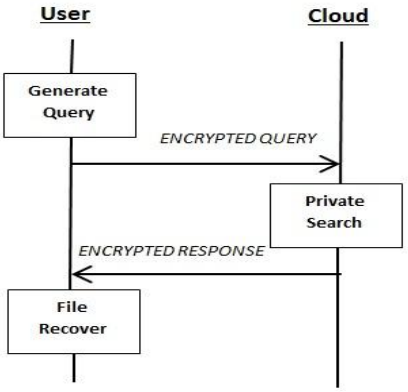

Fig 2 Ostrovsky Protocol

The three algorithms are listed in Fig. 3.

Step 1. The user generates query using *GenerateQuery* algorithm. If the user chooses a word that is part of the dictionary, it is represented as a 1 in the query; otherwise, it is represented as a 0 in the query. The query might look as {*E(0),E(1),E(1),E(0),E(0),E(0)*…} where *E(1)* indicates encrypted value of 1 and *E(0)* indicates encrypted value of 0. User encrypts query using Paillier cryptosystem and sends it to the cloud. Encryption is performed using the public key of the organization.

GenerateQuery (run by user)

for *i*=1 to *d* do
  if the *i*-th keyword in the dictionary *Dict[i]* is chosen then
    Q[i] = 1 else *Q[i]* = 0
    encrypt *Q[i]* with the user's public key

PrivateSearch (run by cloud)

for each file F_j in the cloud do
  for *i*=1 to *d* do
    $c_j = \Pi_{Dict[i] \in Fj} M[i, k];$
    $e_j = c_j^{|F_j|}$

Multiply (c_j, e_j) many times in a compact buffer

FileRecover (run by user)

Decrypt the buffer to obtain plaintext pair (c'_j, e'_j)
if $c'_j = 0$ then recover file content with e'_j / c'_j

Fig 3 Ostrovsky Algorithms

Step 2. The cloud runs the *PrivateSearch* algorithm and returns an encrypted buffer as response to the user. It generates occurrence-content pairs called (*c*,*e*) pairs as part of this algorithm. In this algorithm, c_j indicates the presence of ranked keywords in file F_j.

Step 3. The user runs the *FileRecover* algorithm to recover the files from the response buffer. The user divides the content *e* by occurrence *c* to obtain the matched files.

Issues in Ostrovsky Protocol

- Ostrovsky protocol suffers from the problem of lack of aggregation of queries. Although it ensures privacy by using Paillier cryptosystem, it does not lower the costs incurred by the customers of the cloud.

4.2 Cooperative Private Searching (COPS) Protocol

COPS protocol was proposed by Qin Liu et al. in [4]. It introduced an aggregation and distribution layer (ADL) between the users and the cloud. ADL responsibilities include aggregation of queries from the users and distribution of results from the cloud to the users. The system model is shown in Fig. 4 with a single ADL within the boundary of an organization.

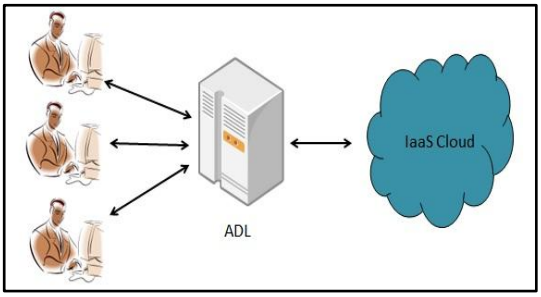

Fig 4 COPS System Model

If the users' queries contain common keywords, it leads to lowered cost since the queries are aggregated by the ADL. Even in the scenario of no common keywords among users' queries, the merging of queries helps in considerably lowered number of round trips to the cloud, thereby lowering the overall costs. COPS protocol works as shown in Fig. 5 [4].

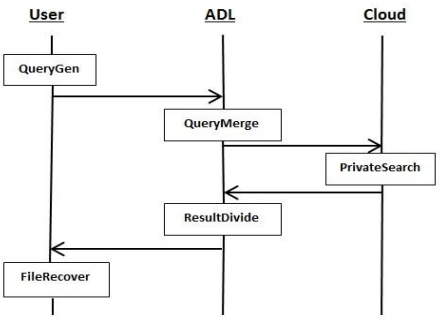

Fig 5 COPS Protocol

Step 1. Individual users generate a query using *QueryGen* algorithm. The query might look as {0,1,1,0,0,0…} where 1 and 0 have the same meaning as in Ostrovsky protocol. Note that the individual user queries are not encrypted before they are submitted to the ADL.

Step 2. The ADL runs the *QueryMerge* algorithm to merge all the user queries and sends a combined encrypted request to the cloud. Encryption is performed using the public key of the organization.

Step 3. The cloud runs the *PrivateSearch* algorithm to find files matching the combined query. The file survival rate in this algorithm is based on the number of mapping times γ and the buffer size β. It sends two encrypted buffers to the ADL, namely file pseudonym buffer and file content buffer. The pseudonym buffer consists of file names replaced with file pseudonyms.

Step 4. The ADL runs the *ResultDivide* algorithm to distribute appropriate files to each user.

Step 5. Individual users run *FileRecover* algorithm to retrieve matched files.

Issues in COPS Protocol

- COPS protocol can send too many results leading to excessive CPU consumption on the cloud. A lot of network bandwidth is required for transferring the response buffers from the cloud.
- ADL introduces delays and the response time is impacted. For users with lower tolerance for delays, multiple ADLs could be deployed within the organizational boundary.

4.3 Efficient Information Retrieval for Ranked Queries (EIRQ) Protocol

EIRQ-Efficient protocol was proposed by Qin Liu et al. in [5]. It shall be referred to as EIRQ in this paper. It is an improvement over COPS protocol. Apart from privacy and

aggregation, its primary motivation is providing the user a means to restrict the number of results.

In what is termed as a *differential query* or *ranked query*, a user is allowed to specify a rank for his query along with the search keywords. Rank 0 indicates that the user is requesting 100% of matched files, rank 1 indicates 75% of files, rank 2 indicates 50% of files and rank 3 indicates 25% of files.

This feature allows the user to restrict the result set to a desired size when there are a large number of files that match the user query. It provides a significant reduction in CPU consumption and network bandwidth usage. Ranked queries therefore lend themselves well to the central premise of pay-as-you-go model of cloud services.

Its system model is same as that used in COPS protocol; however, the algorithms differ. An overview of EIRQ protocol is shown in Fig. 6.

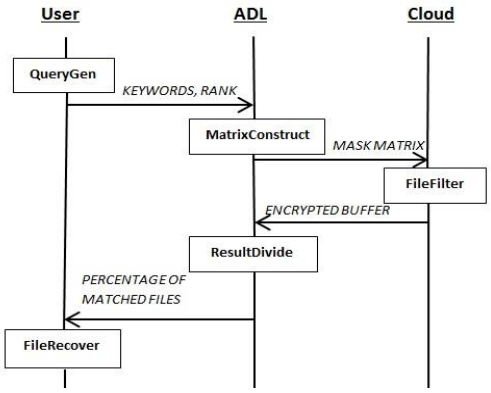

Fig 6 EIRQ Protocol

Step 1. Individual users generate a query using *QueryGen* algorithm. The query is a bit string where each bit is an encryption of 1, if the keyword in the dictionary is chosen; otherwise, it is an encryption of 0. Users also specify a rank along with their query. A notation of {1} is used to indicate an encrypted '1'and {0} to indicate an encrypted '0'. The algorithms are listed in Fig. 7.

Step 2. The ADL runs the *MaskMatrix* algorithm to construct a privacy-preserving mask matrix consisting of all queries. The matrix will consist of d rows and r columns, where d is the number of keywords in the dictionary and r is the highest rank of the queries.

Step 3. The cloud runs the *FileFilter* algorithm to find files matching the merged query. The cloud processes the encrypted query on each file to generate an encrypted c-e pair, and maps it to entries of an encrypted buffer.

For each file F_j, the corresponding c-e pair (c_j, e_j) is generated as follows: the bits in query Q corresponding to keywords in file F_j are multiplied together to form $c_j = \Pi_{Dict[i] \in Fj} Q[i]$, where $Dict[i]$ denotes the *i-th* keyword in the dictionary. File content $| F_j |$ is powered to c_j to form e_j where $e_j = c_j^{| F_j |}$. The cloud sends an encrypted buffer containing the files and corresponding keywords to the ADL.

Step 4. The ADL runs the *ResultDivide* algorithm to distribute appropriate files to each user, based on the ranks and the keywords used by the respective users.

Step 5. Individual users run *FileRecover* algorithm which divides e_j by c_j to retrieve matched files.

QueryGen (run by the users)

for $j = 1$ to d **do**
  if keyword $w_j \in Dict$ is chosen by user i **then**
    $Q_i[j] = 1$
  else
    $Q_i[j] = 0$

MatrixConstruct (run by the ADL with public key pk)

for $i = 1$ to d **do**
  set l to be the highest rank of queries choosing $Dict[i]$
  for $j = 1$ to r **do**
    if $j \leq r - l$ **then**
      $M[i, j] = E_{pk}(1)$
    else
      $M[i, j] = E_{pk}(0)$
adjust mapping times γ and buffer size β so that file survival rate is 1

FileFilter (run by the cloud)

for each file Fj stored in the cloud **do**
  for $i = 1$ to d **do**
    $k = j \bmod r$;
    $c_j = \Pi_{Dict[i] \in Fj} M[i, k]$;
    $e_j = c_j^{| F_j |}$
map (c_j, e_j) γ times to a buffer of size β

ResultDivide (run by the ADL with private key sk)

decrypt response buffer using *sk*
for $i = 1$ to n **do**
  for $j = 1$ to d **do**
    if $Q_i[j] = 1$
     distribute the files F to user i using rank of the query

Fig 7 EIRQ Algorithms

Issues in EIRQ Protocol

- Different users might have different tolerance to delays introduced by the ADL. A potential solution mentioned in the supplemental file pertaining to [5] is the provision of a timer value along with the user query. User *i* could send a timer value of T_i along with the query. ADL would wait no longer than the shortest time T_i of all the queries. At the expiry of shortest time T_i, ADL generates the mask matrix of all queries received so far and sends it to the cloud.

5. COMPARISON OF SECURE SEARCH PROTOCOLS

We present a comparison of Ostrovsky, COPS and EIRQ protocols by comparing their features and performance in a pay-as-you-go cloud.

5.1 Features

In Table 1, the features of secure keyword search protocols are compared vis-à-vis the basic keyword search without any encryption. An 'X' indicates absence of the feature.

Table 1 Features of Search Protocols

Feature	*Keyword*	*Ostrovsky*	*COPS*	*EIRQ*
Encryption	X	✓	✓	✓
Aggregation	X	X	✓	✓
Ranked Queries	X	X	X	✓

5.2 Performance

In the basic keyword search protocols that are used in the cloud in general, the computation cost and communication cost increases linearly as shown in Fig. 8 [4]. This linear increase in complexity is highly undesirable for organizations which are cost-sensitive.

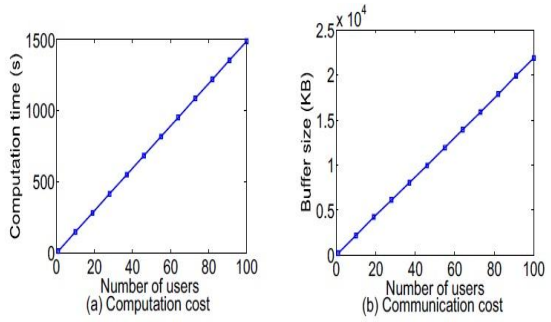

Fig 8 Linear Complexity of Basic Search Protocols

The secure search protocols are compared on the basis of following performance metrics:

- **Computation Cost:** Computation cost is determined by the CPU computations performed on the cloud while searching for the files matching the user queries.
- **Response Buffer Size:** Response buffer size is the size of the response buffer that is transferred from the cloud to the ADL.
- **Communication Cost:** Communication cost is the cost incurred in transferring the response buffer from the cloud to the ADL. The reduction in response buffer size leads to a corresponding reduction in communication cost.

The comparison graphs are from the experiments mentioned in [4] and [5]. These experiments involved MATLAB simulations and deployments on an Amazon EC2 instance.

5.3 Computation Cost

Assume that *n* is the number of users, *t* is the number of files stored on the cloud and *d* is the number of keywords in the dictionary. In Table 2, the computation cost is compared across the three search protocols.

Table 2 Computation Cost of Search Protocols

Metric	*Ostrovsky*	*COPS*	*EIRQ*
Computation	$O(n.t)$	$O(t+d)$	$O(t)$

It should be noted that the size of files does not have any impact on the performance of these protocols because the cloud does not process the contents of the file. The cloud merely processes the user queries based on the keywords associated with each file.

The computation cost is compared between COPS and EIRQ is shown in Fig. 9. Since the Ostrovsky protocol incurs a linear increase in computation cost, it is not included in this figure.

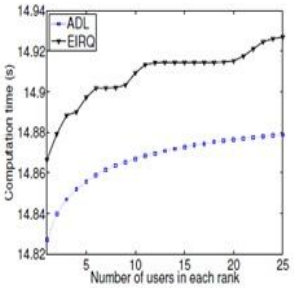

Fig 9 Computation Cost of COPS (ADL) and EIRQ

5.4 Response Buffer Size

The response buffer size of Ostrovsky, COPS and EIRQ is show in Fig. 10 [6]. It shows a dramatic decrease in the EIRQ buffer size as a result of ranked query feature.

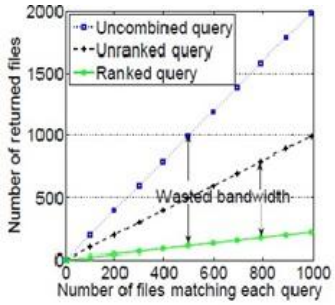

Fig 10 Response Buffer Size of Ostrovsky (Uncombined query), COPS (Unranked query) and EIRQ (Ranked query)

5.5 Communication Cost

The communication cost is driven by response buffer size. It can be inferred from Fig. 10 that the communication cost is lowest in EIRQ protocol when compared to Ostrovsky and COPS protocols.

Another metric related to communication cost is the transfer time from the cloud to the ADL. When deployed on a cloud instance on Amazon EC2, it was confirmed in [5] that the transfer time is lowest with EIRQ protocol.

6. CONCLUSIONS AND FUTURE WORK

This paper presented a description and comparison of Ostrovsky, COPS and EIRQ protocols currently available for use by the customers of pay-as-you-go clouds. EIRQ protocol is the latest among these protocols and it addresses the issues of privacy, aggregation, CPU consumption and network bandwidth usage.

The following aspects of EIRQ could be improved in the future:

1. A "greener" version of EIRQ protocol can be developed to decrease the energy overhead as mentioned in [5].
2. Currently, the ranking of the files on the cloud is determined by the highest rank of the queries matching that file. Since this ranking is not flexible, a more sophisticated ranking system could be developed by providing variable weights to the relevance attributes of each file.
3. The matrix constructed by *MaskMatrix* algorithm has a row for each keyword in the organization's dictionary. It could pose a scalability problem for organizations which have dictionaries with thousands of keywords. A scalable version of this algorithm can be proposed to compress the size of this matrix.

REFERENCES

[1]. P. Mell and T. Grance, "The nist definition of cloud computing," NIST Special Publication 800-145, Sep 2011.
[2]. P.Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in Proc. of EUROCRYPT, 2009.
[3]. R. Ostrovsky and W. Skeith, "Private searching on streaming data", Journal of Cryptology, Volume 20:4, pp. 397-430, October 2007.
[4]. Q. Liu, C. Tan, J. Wu, and G. Wang, "Cooperative private searching in clouds", Journal of Parallel and Distributed Computing, 2012.
[5]. Q. Liu, C. C. Tan, J. Wu, G. Wang, "Towards Differential Query Services in Cost-Efficient Clouds", IEEE Transactions on Parallel and Distributed Systems, Vol. PP No. 99, Year 2013.
[6]. Q. Liu, C. C. Tan, J. Wu, G. Wang, "Efficient information retrieval for ranked queries in cost-effective cloud environments", Proc. of IEEE INFOCOM, 2012.

BIOGRAPHIES

V. Anand, PMP, received M.S. degree in Electrical Engineering from Southern Illinois University at Edwardsville, US. He has over fourteen years of experience in software development in companies such as Comsat, Hughes Network Systems, ACS and Oracle. His interests include cloud computing and non-relational databases. He is currently pursuing M.Tech. degree in CSE from MJCET, Hyderabad. He is a student member of IEEE.

Dr. Ahmed Abdul Moiz Qyser is the Professor and Head of CSE Department at MJCET, Hyderabad. He received Ph.D. in CSE from Osmania University, Hyderabad. He has over seventeen years of teaching experience in the areas of cloud computing, software engineering, operating systems and relational databases. He has trained software teams in companies such as Satyam, Wipro and Qualcomm. He also trains government officials at Engineering Staff College of India, Hyderabad. He is a member of ACM, ISTE and CSI.