An Efficient and Privacy-Preserving Multi-Keyword Ranked Search over Encrypted Cloud Data

Priyatha Joji Abraham, Tina Philip

Department of Computer Science, San Jose State University

priyathajoji@gmail.com, talk2tinaphilip@gmail.com

Abstract – **Development in cloud computing encouraged the data owners to outsource their complex data management systems from their local sites to public cloud for economic savings and easy maintenance. However, as the cloud customers and the cloud server are not in the same trusted domain, the outsourced data may be under the exposure to the risk. Corresponding data owners lose direct control of these data once the sensitive data are outsourced to a remote cloud. Consequently, sensitive data such as emails, personal health records, financial transactions and government confidential files has to be encrypted before outsourcing for preserving data privacy against CSP. In chapter I of this paper, we provide a general introduction about the data privacy and encryption as well as multi keyword ranked search over encrypted cloud (MRSE) data. We include the architecture the efficient similarity measures of "coordinate matching", "inner product similarity" used in rank retrieval. Chapter II defines a system model and a corresponding threat model. Chapter III illustrates the privacy requirements for MRSE framework. In Chapter IV, the proposed MRSE schemes based on the similarity measure of "coordinate matching" and secure inner computation while meeting different privacy requirements in two different threat models would be discussed [1]. In Chapter V, we will discuss the benefits and in Chapter VI, current industrial practice. Chapter VII explains related works. Finally, we conclude in chapter VIII.**

Keywords: cloud computing, sensitive data, data privacy, encrypted cloud data, multi-keyword ranked search, coordinate matching, inner product similarity, secure inner computation, threat models

Table of Contents

List of Acronyms

AWS	Amazon Web Services
C	Encrypted document collection in cloud server C= {Ci} where i > 0
CSP	Cloud Service Provider
d	Number of fields for each database record pi.
D_i	Document vector or data vector of ith document.
Di[j]	Document vector of ith document and jth keyword in $\tilde{\mathcal{W}}$
F	Original document set
F$\tilde{\mathcal{W}}$	Ranked-id list of top k documents sorted with similarity
I	Searchable index
IR	Information Retrieval
k`	Number of real top-k documents returned by cloud server.
KMS	Key Management Service
kNN	k-Nearest Neighbor
ℓ	Security parameter
Mi	Invertible Matrix i, i > 0
MRSE	Multi-Keyword Ranked Search over Encrypted Cloud data
n	Set of distinct keywords in query set
pi	Database vector
P_k	Precision of top-k documents
Qj	Query vector of j^{th} keyword
r	Random number assigned during trapdoor generation in MRSE I
rQ	Query vector associated with a random number in trapdoor generation
S	Randomly generated split query vector in Setup step.
SK	Symmetric Key
T	Search request received from user
T $\tilde{\mathcal{W}}$	Trapdoor generated for search request $\tilde{\mathcal{W}}$
U	Number of dummy keywords inserted in data vector in MRSE II
V	Randomly selected keywords from U in MRSE II
W	Dictionary
$\tilde{\mathcal{W}}$	Query keyword set in trapdoor, subset of dictionary

I. INTRODUCTION

A. Need of Data Privacy and Encryption

Data privacy and confidentially are two significant topics in cloud computing. When the data owners outsource their sensitive data to a commercial public cloud server, there are high chances for it to be vulnerable. Although CSPs provide data security through firewalls, virtualizations, they do not possess full protection in reality. Therefore, sensitive data such as emails, personal health records, financial transactions and government confidential files have to be encrypted by data owners before outsourcing to the commercial public cloud [1]. Encryption on sensitive data before outsourcing can thus preserve data privacy against CSP.

B. Multi-keyword Ranked Search

Multi-keyword ranked search is popular in traditional IR systems. However, the traditional system obsoletes data utilization based on plaintext keyword search. Considering the large number of data users and documents in the cloud, it is necessary to support multi-keyword query and ranked lists of results in the order of their relevance to meet the effective data retrieval need.

C. Privacy requirement

Our scheme is designed to meet the privacy requirement as well as impede the ability of cloud server from learning additional information from index and trapdoor. Cloud server should be retrieving only search results.

For data privacy, the data owner can use symmetric key cryptography to encrypt the data before outsourcing and successfully prevent the cloud server from learning the outsourced data. In Symmetric Key cryptography, same cryptographic keys are used for both encryptions of plaintext and decryption of cipher text.

For index privacy, the cloud server should not deduce any association between keywords and encrypted documents from the index and therefore, the searchable

index should be constructed to prevent the cloud server from performing such kind of association attack on the given dataset.

D. Current similarity measures chosen

Coordinate matching: Gathering as many matches as possible, is an efficient similarity measure to capture the relevance of data documents to the search query [1].

Inner product similarity: Quantifying the number of query keywords appearing in a document, to evaluate the similarity measure of that document to the search query [1].

E. Challenges of the current system

Searchable encryption cannot accommodate high service-level requirements like system usability, user searching experience and easy information discovery. Applying coordinate matching in encrypted cloud data search system remains a very challenging task because of the inherent security and privacy obstacles like data privacy, index privacy, keyword privacy, and many others. Current system does secure ranked search over encrypted data problem, but it works only for queries consisting of a single keyword. Boolean keyword searchable encryption (conjunctive keyword search) incurs large communication and computation overhead.

Moreover, both of these methods retrieves only exact keyword matches to the user and not any approximation values of keywords incase user make any typo-error.

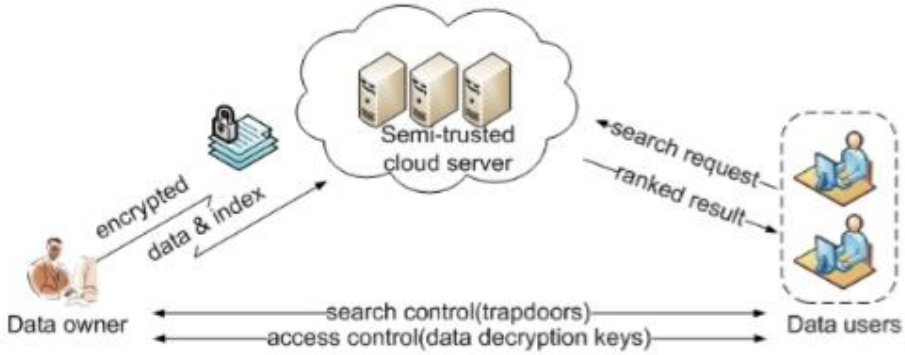


Fig. 1: Architecture of search over encrypted cloud data [1]

II. PROBLEM FORMULATION

A. System model

As illustrated in Fig.1, data hosting service in cloud consists of three different entities namely: the data owner, the data user and the cloud server. The data owner may be an individual or an enterprise, who wishes to outsource a collection of documents F = (F1, F2… Fn) in encrypted form C = (C1, C2… Cn) to the cloud server and still preserve the search functionality on outsourced data. To enable multi-keyword ranked search capability, the data owner constructs a searchable index I from the original dataset F. Both the index I and the encrypted document collection C are outsourced to the cloud server. To securely search the document collection for t keywords, the authorized data user uses search trapdoor distributed by the data owner, using search control mechanisms like broadcast encryption that generates the search request to the cloud server. Once the cloud server receives request T from the data user, it performs a search based on the stored index I and returns a ranked list of encrypted documents. The data user then uses the secret key S, securely obtained from the data owner, to decrypt received documents to original view. Search results are ranked according to some ranking criteria like coordinate matching to improve the document retrieval accuracy. The data user may send an optional number k along with the trapdoor T so that the cloud server only sends back top-k documents that are most relevant to the search query. Finally, the access control mechanism is employed to manage decryption capabilities given to users.

B. Threat model

We assume an "honest-but-curious" model for the cloud server. The cloud server is honest, that is, it is always available to the data user and it correctly follows the designated protocol specification, and it provides all services that are expected. The curious cloud server may try to perform some additional analysis to breach the confidentiality of the stored data. It may try to infer and analyze data in its storage and message flows received during the protocol so as to learn additional information. Based on the knowledge of the cloud server, there could be two threat models:

Known Ciphertext Model - In this model, the cloud server is supposed to only know encrypted dataset C and searchable index I, both of which are outsourced from the data owner.

Known Background Model - In this model, the cloud server is supposed to possess more knowledge than what can be accessed in the known ciphertext model. Such information may include the correlation relationship of given search requests (trapdoors), as well as the dataset related statistical information. As an instance of possible attacks in this case, the cloud server could use the known trapdoor information combined with document/keyword to deduce/identify certain keywords in the query.

III. MRSE FRAMEWORK AND PRIVACY REQUIREMENTS

A. *MRSE Framework*

The MRSE system consists of four algorithms as follows [1].

• Setup (I^l) : Data owner outputs a symmetric key as SK by taking a security parameter ℓ as input.

• BuildIndex (F, SK), Data owner builds a searchable index I which is encrypted by the symmetric key SK and then outsourced to the cloud server dependent on the dataset F. Once we built the index, the document collection can also be independently encrypted and then outsourced to the server.

• Trapdoor ($T\mathcal{W}^{\tilde{}}$) A trapdoor $T\mathcal{W}^{\tilde{}}$ is generated depending on the t keywords of $\mathcal{W}^{\tilde{}}$, search query keyword set issued by user.

• Query ($T\mathcal{W}^{\tilde{}}$, k, I) When the cloud server receives a query request as ($T\mathcal{W}^{\tilde{}}$, k), it performs the ranked search on the index I with the help of trapdoor $T\mathcal{W}^{\tilde{}}$, and finally returns $F\mathcal{W}^{\tilde{}}$, the ranked id list of top-k documents sorted by their similarity with $\mathcal{W}^{\tilde{}}$ [1].

B. Privacy Requirements Established

For establishing privacy in MRSE framework, a set of strict privacy necessities have been used such as data privacy, index privacy, keyword privacy, Trapdoor unlinkability and Access Pattern.

For data privacy, a symmetric key cryptography is used by the data owner to encrypt the data that has been outsourced and effectively prevent the server from intruding into these outsourced data

Then there is index privacy, to prevent cloud server from learning the major content of the documents. The server should not be able to infer any associations between the keywords and encrypted documents from its index. Searchable index built should be capable of inhibiting such association attacks.

Keyword privacy explains that keywords specified by the corresponding trapdoor must be hidden from the cloud server. Trapdoor is generated in a cryptographic way to protect the query keywords. However, cloud server is capable of performing statistical analysis over search result such as finding the documents that matches the search term and then identify the keywords with maximum likelihood estimate. This makes it easy to reverse-engineer the keyword for cloud server.

Trapdoor generation function used to generate trapdoors to users must be a randomized one instead of being deterministic. This method disables the cloud server from gathering relations between any given trapdoor. On the contrary, a deterministic function allows the cloud server to accumulate search frequencies of different search request as the trapdoor generated would be identical for similar search requests. By introducing non-determinacy in trapdoor generation function we ensure that trapdoor unlinkability is maintained.

The final privacy requirement is access pattern in a ranked search. A search result returns a ranked list of documents based on its relevance with respect to the search query and access pattern is the sequence of search results. For instance, given a query keyword set $\mathcal{W}$ an attacker does not learn its keyword, but knows which are the documents that

contains the keyword in query set. For search efficiency concerns, we are not hiding the access pattern.

IV. PROPOSED MRSE SCHEMES

For efficient ranked search, the proposed MRSE scheme is an efficient similarity measure of coordinate matching by inner product similarity while preserving strict system-wise privacy in the cloud computing paradigm. In this similarity measure we use queries as well as documents are represented as vectors in a high dimensional space.

Given a query vector Q[j] $\in$ {0, 1} consists of keyword W_j from query keyword set in trapdoor $\tilde{\mathcal{W}}$ and set of document vectors Di[j] $\in$ {0, 1}. We rank the documents by computing a similarity measure between the query and each document vectors and compares an angle between them using the inner product or dot product of them. i.e., **sim (Di, Q) = Di.Q**.

As we require relevance ranking, cloud server is allowed to compare the similarity of different documents to the query. However, our aim is to preserve strict privacy of system and disable cloud server to learn any additional information. For that we shouldn't reveal the document vector Di, query vector Q and their inner product Di.Q to server.

For secure computation we use **secure inner product** computation - for finding k nearer database neighbors to the query point.

We propose two MRSE schemes based on the similarity measure of "coordinate matching" while meeting different privacy requirements in two different threat models [1].

A. MRSE I: Privacy-Preserving Scheme in Known Cipher-text Model

Secure kNN computation is a scheme adapted from a secure k-NN technique where k nearest database records are selected by computing the Euclidean distances between a database vector pi to a query vector q. Thus the secret key generated contains vector S of length (d + 1) - bit and two invertible matrices as {M1, M2} of size (d + 1) × (d + 1), where d is the number of fields for each record pi.

To fit this computation in MRSE framework we have to modify the data structure such that we use MRSE using inner product similarity instead of using a Euclidean distance. As we are using multi-key word search query vector Qi is used instead of q where *i* is an element of a natural number.

In MRSE I scheme, we assign randomness that increases the difficulty for the cloud server to discover the relationship among the received trapdoors by generating a new random number t to the extended dimension in each query vector. This could obscure the document frequency such that the keywords are not re-identified by server. Randomness is also assigned to data vector by inserting dummy keywords to it.

To retrieve ranked search with multi-keyword we use the following scheme:

1. Setup: Data owner randomly generates a (n + 2)-bit vector S and two invertible matrices {M1, M2} of dimension (n + 2) × (n + 2) where n is the set of distinct keywords in query set. The secret symmetric key SK is in the form of a 3-tuple as {S, M1, M2} [1].

2. BuildIndex (F, SK): For every document Fi in database, data owner creates a binary data vector Di where each binary bit Di [j] represents whether the corresponding j^{th} keyword of $\tilde{\mathcal{W}}$appears in the document Fi. This is repeated for every plain text sub index. In secure kNN computation, $(n+1)^{th}$ entry in vector Di is set to 1 during dimension extending. Ultimately, a sub-index is built for every encrypted document C ={Ci}, i >0 on cloud server.

3. Trapdoor ($T\tilde{\mathcal{W}}$): This algorithm generates a corresponding trapdoor $T\tilde{\mathcal{W}}$. If a server is interested in t keyword of input search query $\tilde{\mathcal{W}}$, then a query-binary vector Q is generated where each bit Q[j] indicates whether keyword subset is an element of $\tilde{\mathcal{W}}$. For example, query $\tilde{\mathcal{W}}$= “Samsung galaxy” consists of 2 distinct keywords “Samsung” and “galaxy”, then query binary vector Q[Samsung] = 1and Q[galaxy] =1.Query vector is generated only for the distinct search query keywords. Q is then scaled by a random number r ≠ 0 as rQ and finally extended to n+2 dimension vector as Q where last dimension is set to another random number t and hence Q = (rQ, r, t) [1].

4. Query ($T_{\tilde{\mathcal{W}}}$, k, I): After generating trapdoor $T_{\tilde{\mathcal{W}}}$ user issues a query to cloud server consisting of the trapdoor generated in previous step, top k documents that should be returned by server based of relevance for ranked search on the index I. For each document Fi in document set F, similarity scores are computed, sorted in descending order and the top k ranked-id list $F_{\tilde{\mathcal{W}}}$ is returned by cloud server.

In the steps like BuildIndex or Trapdoor, the generation procedure of each sub-index or trapdoor involves two multiplications of a (n + 2) × (n + 2) matrix and a (n + 2)-dimension vector. In the query, the final similarity score is computed through two multiplications of two (n+2)-dimension vectors.

The index privacy is well protected if the secret key SK is kept confidential since such vector encryption method has been proved to be secure in the known cipher text model [1].

Moreover, two totally different trapdoors are generated for the same query $\mathcal{W}$ by introducing randomness using random numbers r and t keywords of $\tilde{\mathcal{W}}$. For e.g.: if user searches for query "gravity on earth" subsequently, then trapdoor generated for the same query at two instances would be totally different. This nondeterministic trapdoor generation can guarantee the trapdoor unlinkability which is an unsolved privacy leakage problem in related symmetric key based searchable encryption schemes because of the deterministic property of trapdoor generation [1].

B. MRSE II: Privacy-Preserving Scheme in Known Background Model

If cloud server understands the background information of data outsourced and deduce relationship of two trapdoors, certain keyword privacy would not be guaranteed anymore by the MRSE I scheme which is possible in Known background model. Hence an advanced MRSE II scheme is proposed to prevent such scale analysis attack that breaks down the keyword privacy.

More dummy keywords are inserted into every data vector Di instead of a single dummy keyword such that the vectors are extended to (n + U +1)-bit vector instead of (n+2) as in MRSE I where U is the number of dummy keywords inserted.

1. Setup(1^n) The data owner randomly generates a (n + U +1)-bit vector as S and two (n + U +1) × (n + U +1) invertible matrices {M1, M2} [1].
2. BuildIndex (F, SK): The (n+j+1)-th entry in Di where j ∈ [1, U] is set to a random number during dimension extending [1].
3. Trapdoor: Randomly select V out of U dummy keywords to generate the trapdoor. Also corresponding entries in query Q is set to 1 [1].
4. Query ($T\tilde{\mathcal{W}}$, k, I): Eventually cloud server computes the similarity score f for each document Fi in document set F. Then it returns top k documents to the user.

Consequently, MRSE II scheme is more secure against scale analysis attack, and provides various expected privacy guarantees within the known cipher text model or the known background model [1].

V. BENEFITS

A. Performance Benefits

1. Precision keeping and privacy-preserving - data privacy, index privacy and search privacy like keyword privacy.
2. Highly efficient due to low overhead on computation and communication.
3. Secure against attacks such as scale analysis attack.

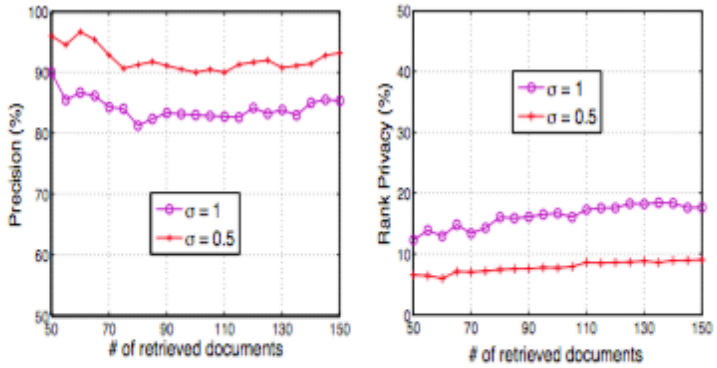
B. Performance Analysis

1. Precision and Privacy

During a ranked search when top-k documents are returned by the cloud server based on similarity scores of data vectors to query vector, a few actual top-k relevant documents for the query may be excluded. Since we are inserting dummy keywords into data vectors the original similarity scores can be decreased or the similarity scores of some documents out of the real top-k are increased. To

measure that phenomenon, we introduce precision of returned result set. Precision is evaluated using the measure P_k= k'/k where k' is the number of real top-k documents returned by cloud server.

For preserving privacy, we can cover the rank order of retrieved documents as search results cannot be hidden. And this scheme provides a balance parameter for data users to satisfy their different requirements on precision and rank privacy [1]. To receive high precision, standard deviation must be small (Refer Fig.3). But having small standard deviation can leak the rank privacy by learning the access patterns by server. Hence there is a tradeoff between precision and rank privacy.



(a) (b)

Fig. 3: With different choice of standard deviation σ for the random variable ε, there exists tradeoff between (a) Precision, and (b) Rank Privacy.

[1]

2. Efficiency

i. Index Construction:

For each document Fi, data owners send searchable sub index I_i to the server. In this sub index Ii, first step is to map the keyword set extracted from the document F_i to a data vector D_i, followed by encrypting every data vector for each document F_i in the dataset. Time cost to build the whole index is linearly proportional to the size of dataset. I.e., Time complexity = O(|W|)

where W is the whole dictionary or dataset. Time cost of building each sub index is fixed

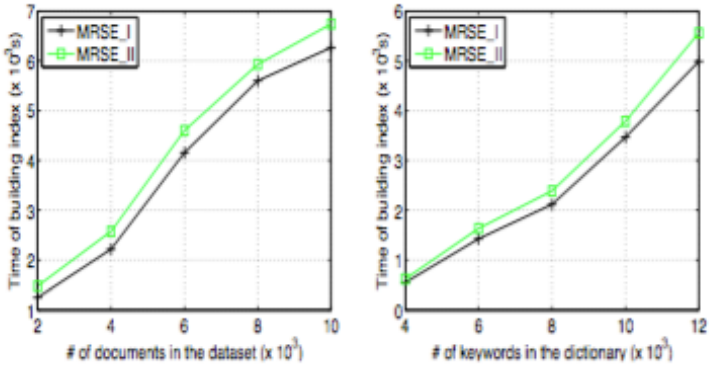


Fig. 4: Time cost of building index. (a) For the different size of dataset with the same dictionary, n = 4000. (b) For the same dataset with different size of dictionary, m = 1000.

[1]

ii. Trapdoor Generation:

Time to generate a trapdoor depends upon the number of keywords in the dictionary. For every trapdoor generation there is 2 invertible matrices multiplication with a split query vector and dimension of both of these can vary in both MRSE schemes. The figure below illustrates that trapdoor generation cost in the MRSE II scheme is about 20 percentages greater than that in the MRSE I scheme.

This explains that trapdoor generation does not depend upon number of query keywords. This gives a substantial advantage to multi-keyword searchable encryption compared to the single key encryption or Boolean encryption techniques.

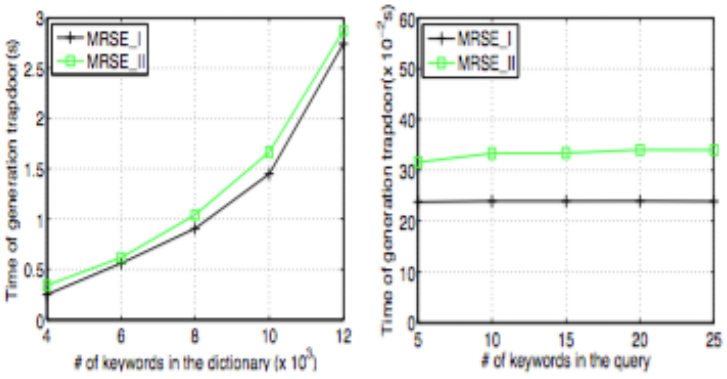


Fig. 5: Time cost of generating trapdoor. (a) For the same query keywords within different sizes of dictionary, t = 10. (b) For different numbers of query keywords within the same dictionary, n = 4000.

[1]

iii. Query:

In the query execution step happening at cloud server, similarity scores are computed and ranked for all documents in the dataset. In this proposed scheme, computation and communication cost in the query has nearly constant overhead while increasing the number of query keywords. From Fig. 6, we can understand that number of keywords in the query has only little influence on query time and query time depends majorly on the number of documents in data set.

Thus when the number of query keywords are increased in search query, this proposed MRSE schemes presents nearly constant overhead compared to single and Boolean encryption schemes.

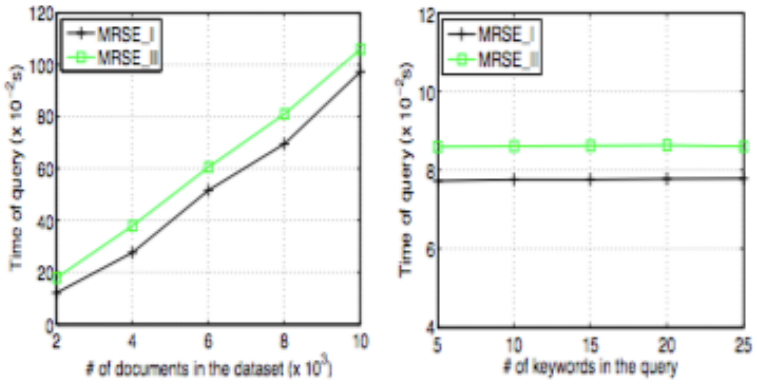


Fig. 6: Time cost of query. (a) For the same query keywords in different sizes of dataset, t = 10. (b) For different numbers of query keywords in the same dataset, m = 1000. [1]

VI. CURRENT INDUSTRY PRACTICE

A. Encryption :

In Amazon Web Services (AWS), client-side encryption refers to encrypting data before sending it to Amazon S3 using an AWS KMS-managed customer master key or a client-side master key. On the client side, a KMS-managed customer master key (CMK) is used to encrypt data before uploading to Amazon S3.

a) Object upload: Client first sends a request to AWS KMS for a key to encrypt the object data using customer master key ID. AWS KMS returns a randomly generated data encryption key of two versions.

i) A plain text version that the client uses to encrypt the object data.

ii) A cipher blob of the same data encryption key that the client uploads to Amazon S3 as object metadata.

b) Downloading an object: Encrypted object which could be a file is downloaded by client from Amazon S3 along with the cipher blob version of the data encryption key

stored as object metadata. This cipher blob is then sent to AWS KMS to get the plain text version of the same, so the object data can be decrypted.

B. Searching:

In Amazon cloud search, a search domain is initially created and then uploads the data to make it searchable in JSON or XML format and then the resources are automatically provisioned. Search parameters are provided by the user and enables fine tune search such as autocomplete suggestions, free text search, Boolean search, Prefix search, Range searches, relevance –ranking and query-time rank expressions, field weighting, geospatial search etc.

VII. RELATED WORKS

Currently organization practice searchable encryption schemes. Searchable encryption treats encrypted data as documents and allows a user to securely search through a single keyword and retrieve documents of interest [1]. Searchable encryption focuses on Single keyword searchable encryption or Boolean keyword searchable encryption and rarely differentiates the search results.

Searchable encryption cannot accommodate high service-level requirements like system usability, user searching experience and easy information discovery [1].

A. Single keyword searchable encryption

Current system does secure ranked search over encrypted data problem only for queries consisting of a single keyword. It was proposed to encrypt each word in the document independently. This method has a high searching cost due to the scanning of the whole data collection word by word [4].

Single keyword usually builds an encrypted searchable index such that its content is hidden to the server unless it is given appropriate trapdoors generated via secret key. To

enrich search functionalities, conjunctive keyword search over encrypted data have been proposed.

B. *Boolean keyword searchable encryption*

Boolean keyword searchable encryption (conjunctive keyword search) incurs large communication and computation overhead. Conjunctive keyword search returns "all-or-nothing", which means it only returns those documents in which all the keywords specified by the search query appear; disjunctive keyword search returns undifferentiated results, which means it returns every document that contains a subset of the specific keywords, even only one key- word of interest.

In short, none of existing Boolean keyword searchable encryption schemes support multiple keywords ranked search over encrypted cloud data while preserving privacy [1].

Furthermore, most of these schemes are built upon the expensive evaluation and their top-k retrieval is also loosely connected.

VIII. CONCLUSION

A. Summary

We define and solve the problem of multi-keyword ranked search over encrypted cloud data, and establish a variety of privacy requirements. We choose the efficient similarity measure of "coordinate matching" and use "inner product similarity" to quantitatively evaluate such similarity measure [1].

Furthermore, we provide two improved MRSE schemes to achieve these privacy requirements in two different threat models thereby reducing the overhead on computation and communication.

B. Challenges and Future works

Main limitation of this system is it doesn't tolerate human typo errors by providing approximate string matching. Only exact keyword matches are returned to the user and typo

errors will not be tolerated. Our Future work would be focusing on implementing this approximate string matching.

Protecting identity privacy of user is a very important factor and user's identity (ID) is not kept hidden in this system. Due to this, whoever puts the data on Cloud Service Provider was known. This may be risky in some situations where identity of user needs to be securely protected. Moreover, we also plan to enhance the system by providing a solution for easy discovery of user search query words in result in our future works.

IX. REFERENCES

[1] N. Cao, C. Wang, M. Li, K. Ren and W. Lou, "Privacy-preserving multi-keyword ranked search over encrypted cloud data," in *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 1, pp. 222-223, Jan 2014. Available: http://ieeexplore.ieee.org.libaccess.sjlibrary.org/document/6674958/

[2] W. Zhang, Y. Lin, S. Xiao, J. Wu and S. Zhou, "Privacy Preserving Ranked Multi-Keyword Search for Multiple Data Owners in Cloud Computing," in *IEEE Transactions on Computers*, vol. 65, no. 5, pp. 1566-1577, May 1 2016. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7130597&isnumber=7447853

[3] L. Chen, X. Sun, Z. Xia and Q. Liu "An Efficient and Privacy-Preserving Semantic Multi-Keyword Ranked Search over Encrypted Cloud Data," *International Journal of Security and Its Applications*," 2014 Available: http://dx.doi.org/10.14257/astl.2013.31.58

[4] C. Chen *et al*., "An Efficient Privacy-Preserving Ranked Keyword Search Method," in *IEEE Transactions on Parallel and Distributed Systems*, 2016 Available: http://ieeexplore.ieee.org.libaccess.sjlibrary.org/document/70919

[5] T. S. Moh and K. H. Ho "Efficient semantic search over encrypted data in cloud computing," *International Conference on High Performance Computing & Simulation (HPCS),* 2014 Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6903711&isnumber=6903651

[6]"Protecting Data Using Client-Side Encryption - Amazon Simple Storage Service", *Docs.aws.amazon.com*, 2016. [Online]. Available: http://docs.aws.amazon.com/AmazonS3/latest/dev/UsingClientSideEncryption.html.