MSSS: Most Significant Single-keyword Search over Encrypted Cloud Data

Raghavendra S, Geeta C M, Shaila K, Rajkumar Buyya, Venugopal K R, S S Iyengar, L M Patnaik

***Abstract*—Cloud Computing is a popular computing technique via the Internet. The data owner outsources local data to the public cloud server to reduce the cost of the data management. Sensitive data has to be encrypted to ensure privacy before outsourcing. All traditional Searchable Symmetric Encryption (SSE) schemes search only over encrypted data through keywords,hence they do not provide effective data utilisation for large dataset files in cloud. In this paper, we propose a Most Significant Single-keyword Search (MSSS), that supports efficient search using a Most Significant Digit (MSD) radix sort. MSD radix sort is simple and faster in sorting array strings. A mathematical model is developed to encrypt the indexed keywords for secure search without the overhead of learning from the attacker/cloud provider. The proposed scheme reduces the computation overhead. Through numerical analysis, it is shown that the MSSS scheme can reduce the computation cost of data on owner side to $O(N_T \times 3)$. The time complexity of search time is reduced to $O(B)$ for bucket size B. The proposed scheme is highly secure and efficient in comparison to the state of the art works.**

***Index Terms*—Keyword Search, Radix Sort, Data Privacy, Searchable Encryption, Cloud Computing.**

I. Introduction

CLOUD computing is a shared pool of computing resource to store or access data from a remote place. Enterprises outsource their data on the cloud. On-demand resource availability and pay-as-use concept has attracted many benefits of new computing model including relief in storage management and global data access [1]. Most of the companies face problem in secure information storage and retrieval on cloud. Data needs to be encrypted before outsourcing to cloud containing sensitive information like email, health records, financial transaction and government documents etc. The cloud provider and unauthorised person can leak the data from the untrusted cloud.

Companies, health care centers and government are outsourcing the documents onto the cloud storage space since they are finding it difficult to maintain the hardware infrastructure on premises. Companies like Amazon, Windows Azure, IBM etc.. provide cloud services based on IaaS(Infrastructure-as-a-Service). Data is encrypted before outsourcing for privacy concerns; the data owner shares the encrypted data with a cloud server and then it is retrieved whenever required. Effective data utilisation is a challenging task for a large number of outsourced data files. Keyword-based search is one of the most popular technique used for searching documents on encrypted cloud data. Keyword search techniques are widely used in plain-text scenarios and the user is allowed to retrieve select files from the storage space.

All traditional Searchable Symmetric Encryption (SSE) (e.g., [2], [3], [4]) schemes allow a user to search on cipher text and securely retrieve the cipher text over encrypted cloud data through keywords without decrypting the files. Boolean keyword search has a main drawback whenever a huge number of documents are involved. The user wants to find matching document for each search request without the pre-knowledge about the encrypted cloud data and wants to scroll through the entire retrieved files, which requires (i) large amount of post-processing when they go through unrelated files resulting in enormous network traffic. (ii) it incurs communication overhead. The above drawbacks can be overcome with top-k single keyword retrieval techniques [5], and single-keyword retrieval techniques [6].

Motivation: In the previous schemes, Boolean and single-keyword search are used to search keywords on encrypted cloud data. The main issue is to reduce the cost of computation without affecting the extracted keyword of the documents to provide security and to select accurate keyword search over encrypted cloud data.

Contribution: In this paper, we have developed a new index building technique known as Most Significant Single-keyword Search (MSSS) Scheme, obtain accurate search results in a short time on the user side and in addition, protect sensitive data from Cloud Service Provider and unauthorised entities. We address these Challenges by our proposed MSSS scheme for secure and efficient single-keyword search. The significance of our contributions are:-

1) The Most Significant Single-keyword Search (MSSS) scheme over encrypted cloud storage data reduces search time over large data sets on cloud.
2) The Most Significant Single-keyword Search algorithm reduces the index generation time to $O(N_T \times 3)$ and supports single-keyword top-k retrieval over encrypted cloud data.
3) A new mathematical model is developed for secure

Raghavendra S, Geeta C M, Venugopal K R are with the Department of Computer Science and Engineering, University Visvesvaraya College of Engineering, Bangalore University, Bangalore, India 560001 e-mail: raghush86@gmail.com, venugopalkr@gmail.com.

Shaila K is with the Department of Electronics and Communication Engineering, Vivekananda Institute of Technology, Bangalore,India e-mail: shailak17@gmail.com.

Rajkumar Buyya is with Grid Computing and Distributed Systems (GRIDS) Laboratory, Department of Computer Science and Software Engineering, The University of Melbourne, VIC 3053 Melbourne, Victoria, Australia e-mail: raj@csse.unimelb.edu.au .

S S Iyenger is with Department of Computer Science and Engineering, Florida International University, USA e-mail: iyengar@csc.lsu.edu

L M Patnaik is with Department of Computer Science and Automation, Indian Institute of Science, Bangalore, India 560012 e-mail: patnaiklm@sristi.cedt.iisc.ernet.in.

encryption search over index, without learning anything about queried keyword from unauthorised entities.

Organisation: The rest of this paper is organised as follows: Related works are discussed in Section 2. Background work with respect to traditional schemes is presented in section 3. Problem statement, system model and design goals are explained in section 4. In section 5, the proposed Searchable Encryption Scheme is developed. Performance Analysis is given in section 6. Conclusions are presented in section 7.

II. LITERATURE SURVEY

We discusses various existing technique focused on secure keyword search on cloud environments. We also identify their strength and weakness. In rest of paper we demonstrate how our scheme overcomes those weaknesses.

Multi-keyword search on encrypted cloud data have been investigated in [6], [7], [8]. It provides security and efficient search by using two thread models, cipher-text model and background model. A secure k-Nearest Neighbor (k-NN) algorithm was implemented in MRSE scheme [6].Orecik *et al.,* [9] proposed an efficient privacy-preserving search over encrypted cloud data that utilises minhash functions to improve the precision rate. The advantages of this scheme are multi keyword search in a single query and effective ranking capability based on term frequency and inverse document frequency.

Ranked searchable encryption schemes are explored in [5], [10], [11], [12] over encrypted cloud data. The secure ranked keyword search schemes are stronger security definition compared to SSE. The Order Persevering Mapping(OPM) technique is used in [5],[10],[12] for ranking the searched file over encrypted cloud data. The OPM technique protects the sensitive score information from the cloud provider. OPM method is highly efficient but they lead to collisions in the network and increases computation cost. One-to-many OPM technique is presented in [10], [12] for secure term frequency. Sun *et al.,* [10] developed a Secure Ranked Semantic Keyword Search (RSS) over encrypted cloud data. A fuzzy solution contributes to search the semantic keyword on encrypted cloud data. The data owner generates a piece of metadata for each file and uploads the encrypted set of metadata and collection of document to the cloud. Semantic search returns exactly matched semantically related files to the queried keyword.

Privacy preserving keyword search schemes are proposed in [3], [9], [13], [14], [15], [16], [17] focusing on security definitions and encryption techniques. The Decisional Diffie-Hellman(DDH) are adopted in [3],[18] for security assumptions. Boneh *et al.,* [3] have defined the mechanism of a public key encryption technique (PKES) with keyword search. The user privacy is not violated and gateway does not learn anything about the encrypted mail. Kumarverma *et al.,* [16] have investigated a new Dictionary and Lingual Keyword based Secure Search method to find exact encrypted cloud data by using multi-lingual search queries, phonetic and also keeping the data integrity of the cloud data storage. The time is reduced to search documents on encrypted cloud data but distribution of dynamic key to enable stronger security.

III. BACKGROUND WORK

Yu et al., [19] have developed Searchable Encryption(TRSE) technique supporting top-k multi keyword extraction from the cloud storage system. Searchable Symmetric Encryption (SSE) technique is used to retrieve encrypted data over cloud. The homomorphic encryption is implemented to rank the searched data. The TRSE technique ensures security for small datasets. The user encrypts and send the cipher-text to the cloud server. The size of the cipher-text is too large. Therefore, the encrypted trapdoor size is too large for communication. The computation overhead on server side is dependent on $tf-idf$ weights to calculate relevance score for each keyword search request.

A. Vector Space Model

The vector space model does not effectively work on large scale datasets. The vector space model matrix involves keywords (w_i), file identifiers ID_i and frequency score (S). The index file takes more and unnecessary storage space. If the frequency is zero then the keyword (w_i) does not appear in the file ID_i but the memory is allocated to store the value zero which increases the search time because of a large number of zeros. In TRSE, the analysis is limited to 1000 keywords (w_i). Homomorphic encryption computes entire matrix including zeros that incurs high computation overhead.

TABLE I
VECTOR SPACE MODEL DATA REPRESENTATION

Terms / File IDs	w_1	w_2	w_3	w_4	w_5	w_6	w_7	w_8
ID_1	1	0	1	0	1	1	1	1
ID_2	3	2	1	2	0	0	2	2
ID_3	1	3	2	0	0	2	0	1
ID_4	0	0	3	0	2	1	2	0

IV. PROBLEM STATEMENT AND SYSTEM MODEL

Consider n files to be outsourced on the cloud server of $honest - but - curious$ model used in most of the SSE scheme. This cloud server should act as an honest follower of the designed protocol. It should be curious enough to infer and analyse the word during message flow to learn additional information. The main objectives are to:

- reduce the secure search time over encrypted cloud data without learning any extra information.
- reduce communication overhead.

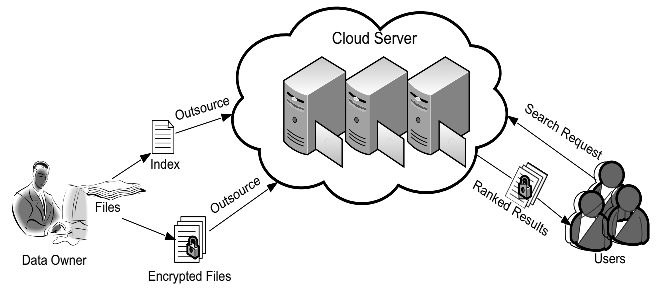

Fig. 1. system Architecture

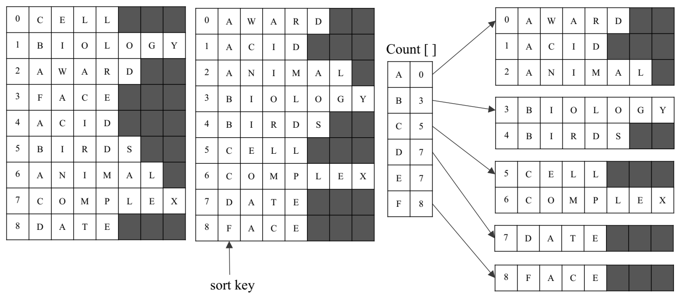

Fig. 2. MSD Sort Process

A. System Model

The architecture of encrypted cloud hosting services is shown in Figure 1. It contains three entities: data owner, data user and cloud server. The System provides most significant keyword search over encrypted cloud data
A collection of n data files $F = (f_1, f_2, f_3, \ldots, f_n)$ to be outsourced onto the cloud server in encrypted form and then provide keyword search services to authorised users. The files extract the m keywords $W = (w_1, w_2, \ldots, w_n)$ from each file to generate encrypted searchable index I'' from F' and stores both the I'' and F' on the cloud.

The Data User is authorised to process multi-keyword search on encrypted cloud data. After conformation of authorisation, the trapdoor t_w of the keyword w_i is generated to send the encrypted search query on the cloud server. When data user submits the search query, the cloud server is searches the index I' and returns the relevant files to the data user. The ranked criteria is used to enhance the file retrieval accuracy of the search result. The data user can reduce the communication cost by sending the optimal value k along with trapdoor t_w and the cloud server sends back the top-k search results.

The third party data storage and retrieval service hosts on cloud server. which is termed as honest-but-curious in our model because they are communicating with both data owners and data users. The storage data may contain sensitive data, and hence the cloud server cannot be fully entrusted in protecting data. They do not delete or modify the user data but try to learn the content of the stored data.

B. Design Goals

The main design goals of the proposed scheme include effective single-keyword ranked search on encrypted cloud data, search result accuracy, security and performance.
Accurate Single-Keyword Ranked Search: To design search scheme for single-keyword queries, accurate similarity results are retrieved and ranked by effective utilisation of outsourced cloud data instead of getting unrelated results.
Security guarantee: To prevent the cloud server to learn the content of the stored data as well as the searched keyword information. It gives stronger security strength compared to existing searchable schemes.

C. Radix Sort

The radix sort is an algorithm to rearrange the string representation process over individual string either in ascending or descending order. It is a non-comparative and a linear sorting string algorithm [20]. The string sorting algorithm sorts data by grouping keys which shares the same significant position and value. The string sorting are of two types: least significant digit (LSD) radix sort and most significant digit (MSD) radix sort [21].

D. Most Significant Digit (MSD)

In MSSS scheme, function 1 focuses on MSD radix sort process. The string representation starts from the most significant digit and moves towards the least significant digit (left most string to right most string for each word). LSD radix sort works the other way around. It is suitable for string like words (variable-length alphabets) and fixed-length integer. Counting sort for one level of buckets to group the keys is used. The indexes file I partitions into R pieces according to the first character and groups the elements with the same character into a bucket. It processes recursively sorting from left to the right of string in each bucket. Finally, all the buckets are concatenated in order. The entire process is shown in Figure 2.

1) Counting sort: After MSD sorting, the string is processed with counting sort algorithm to count the objects of the bucket element [22]. An array A of n most significant elements is taken from the keywords in the range $(1, 2, \ldots, k)$. The counting sort keeps an auxiliary array $C[i]$ range 1 to k initialised to $zero$. The algorithm makes a pass through input array A; for each element i of A, increments $C[i]$ by 1. The iteration is done for n elements of array A with time complexity of $O(n)$ times and updates C. The index j values of C shows the number of times that j appears in A. The next step is to insert each element j with a total of $C[j]$ times in the new list of C' and it computes with complexity $O(k)$. The total time taken for counting sort is of the order $O(n + k)$.

E. Keyword Compution Method

The Score S is calculated based on the frequency of each term in the individual file. The expression for normalised Score calculation is as follows:

$$S = \frac{freq}{maxfreq}. \tag{1}$$

TABLE II
NOTATIONS

Symbols	*Definition*
F	The plain-text document collection to be outsourced as a set of n data files $F = (f_1, f_2, f_3, \ldots, f_n)$.
W	The extracted distinct keywords from the document collection F, a set of m keywords $W = (w_1, w_2, \ldots, w_m)$.
I	The searchable index built from the document collection F, denoted as $(I_1, I_2, \ldots, I_m)$ where each sub-index I_i build from F_i.
t_w	The trapdoor generated for search request of keyword W.
ID_{list}	The set of ranked identifiers of files in F that contains keyword w_i.
$ID(f_i)$	The file identifiers f_i to locate the actual file.
S	Score is calculated by using term frequency TF.
a	ASCII value of alphabets in each letter of the keyword.
$\alpha(w_i)$	Compute result for each extracted keyword from equation-2.
D	Number of Documents.
T	Number of terms in each document.
N_T	Number of rows in index file.
C	Number of columns in index file (i.e., C=3).
$c(i)$	Contains elements in the sort list $(i = 1, 2, \ldots, n)$.
w_i	Individual extracted keyword.
F'	Encrypted n files.
I'	Stored all the computed $\alpha(w_i)$ in the Index
I''	Encrypted I'
$A(j)$	Numbrer of extracted keyword W in an array taken as input for Function 1.

where $freq$ - frequency of each term in a file, $maxfreq$ - maximum frequency after considering all the files in the folder and S - is Score obtained by $\frac{freq}{maxfreq}$. A new mathematical model for encrypting the keyword is given below:

$$\alpha(w_i) = (a_0 x^k + a_1 x^{k-1} + \ldots\ldots + a_n x^{k-n}) \tag{2}$$

$$\alpha(w_i) = \sum_{p=0}^{n} a_b x^{k-p} \tag{3}$$

where x - is a real number and it should be same for both index keyword and queried keyword, k - is a length of the keyword(i.e., if the keyword is $Network$ than the length of the keyword is 7) and p - is the position of the each letter $(0 \leq p \leq n)$ (if the keyword $Network$ position of letter e is 2 and r is 6).

V. PROPOSED MSSS SCHEME

MSSS scheme generates index for optimal Secure multi-keyword search time over encrypted files and reduce the index storage space in the cloud sever. The framework of MSSS scheme involves four functions $Setup$, $BuildIndex$, $TrapdoorGen$ and $SearchResult$.

- $Setup(\lambda)$: The secure input parameter λ generates Public Key(PK) and Secret Key(SK) for the Most significant digit searchable encryption scheme. The data owner distributes Secret Key to the authorised users.
- $BuildIndex(F, PK)$: The collection of files F extracts the unique keyword to construct the searchable index I. Sorting is based on the MSD radix sort technique

Function 1: MSD Radix Sort

```
input    : Extracted Keyword from the text
           $W = (w_1, w_2, \ldots, w_n)$ and most
           significant digit $d$
output   : Index $I$ sort by most significant digit in
           Ascending Lexicographical order
Function: MSD$(W, d)$
1)Take the MSD for the first character of each $w_i$;
2)Sort the $W$ based on the first digit of each keyword
$w_i$ using countingsort();
3)Grouping elements with the same digits into a
bucket $B_i.(i.e., i = 1, 2, \ldots, n)$;
4)concatenate the buckets $(B_1, B_2, B_3, \ldots, B_n)$
together in order;
Procedure countingsort(A[j])
    for $i \leftarrow 1$ to $k$ do
        $C[i] \leftarrow 0$;
    for $j \leftarrow 1$ to $n$ do
        $C[A[j]] \leftarrow C[A[j]] + 1$
    ;
    for $i \leftarrow 2$ to $k$ do
        $C[i] \leftarrow C[i] + C[i-1]$
    ;
    for $j \leftarrow n$ downto $1$ do
        $B[C[A[j]]] \leftarrow A[j]$;
        $C[A[j]] \leftarrow C[A[j]] - 1$;
```

Algorithm 1: Most Significant Multi-keyword Search (MSSS)

```
Initialisation Phase
input    : A set of $n$ Data Files $F = (f_1, f_2, \ldots, f_n)$
output   : Index file generated from extracted
           keyword $I$
Function: BuildIndex$(K, F)$
for $f_i \leftarrow 1$ to $n$ do
    each file $f_i \in F$;
    Scan $F$ and Extract the distinct word in $f_i$,
    denoted as a $W = (w_1, w_2, w_3, \ldots w_n, )$ ;
    Normalised and filter the stopwords from $W$;
    for $j \leftarrow 1$ to $m$ do
        each file $w_i \in W$;
        1) Calculate the Score $S$ for each keyword $w_i$
        according to equation 1;
        2) MSD() to sort the Index $I$;
        3) Compute $\alpha(w_i)$ for each keyword $w_i$
        according to equation 2;
        4) Store the $\langle id(f_i)||\alpha(w_i)||S\rangle$ as an element
        in the posting list of $I'$ ;
    5) Encrypt the Index file $I'$;
    6) Replace $I'$ with $I''$
    return $I''$;
```

and computes keywords according to Equation 2 in the searchable index I'. The searchable index I' also contains frequency based relevance score and file IDs. Finally, the searchable index I' is encrypted into I'' with PK, the output I'' is uploaded to the cloud storage space.

- $TrapdoorGen(I', t_w)$:The data user generates secure trapdoor t_w corresponding to the interested query keyword request Q. the multi-keyword request encrypts into a secure trapdoor t_w to search on the encrypted data I'.
- $SearchResults(I', t_w)$: On receiving the Secure trapdoor t_w. The cloud server computes and returns the matched file IDs and relevant score gets back to the users in the descending order. The top-k matched files is sent back in a ranked sequence based on the relevance score. Top-k files are securely retrieved without learning anything about the search keyword and index I'.

The framework of the Algorithm 1 can be built in two phases: (i) initialisation phase and (ii) retrieval phase. The initialisation phase involves $Setup$ and $BuildIndex$ functions. The Setup function processes on data owner side to generate SK and PK for individual authorised users. BuildIndex function involves operations on plain-text and generates secure searchable index from the plain-text files F. The searchable index $(n \times 3)$ matrix involves extracted keyword, file IDs and score for convenient retrieval of data from the Function 1 and the initialisation phase. For security concerns, most of the work is conducted on the data owner side. The details of the $initialisation$ and $retrieval$ phase is described below:

Initialisation Phase:

- The data owner generates Secret Key(SK) and Public Key(PK) for the $MSSS$ scheme by calling the function $KeyGen(\lambda)$. The data owner shares the secret keySK with the authorised data users to access cloud data files $f_i(1 \leq i \leq n)$.
- The data owner extracts the collection of m keywords W=$(w_1, w_2, w_3, \ldots, w_m)$ from the scanned files where $W = (w_i|1 \leq i \leq m)$ and then filters the stop words from W and term frequency TF. Compute for each file f_i belonging to F, the score S is calculated according to Equation 1. The data owner builds a $(n \times 3)$ matrix to store the extracted keywords in the form $(w_i||ID(f_i)||S)$ into the index file I. W is the Extracted keywords after filtering the stopword, $ID(f_i)$ is the file identifiers for F files and S is a score calculated by normalised term frequency (TF). The index file I is sorted by using MSD Radix sort $Function1$. The explanation is elaborated in session 3. The counting sort function the group elements with the same MSC into a bucket $B_i(i = 1, 2, \ldots, n)$ and then concatenates the buckets $B_i = (B_1, B_2, \ldots, B_n)$ together in lexicographical order. After sorting all the extracted keyword present in the index file I, each keyword is encrypted according to Equation -2. Now the searchable index file I' is partially encrypted.
- The data owner encrypts both the Searchable index file $I' = (v'_i|1 \leq i \leq n)$, where $v'_i = (\alpha(w_i||ID(f_i)||S)$ into I'' and the file $F = (f_1, f_2, \ldots, f_n)$ into $F' = (f'_1, f'_2, \ldots, f'_n)$ with cryptology techniques and then the I'' and F' is outsourced to the cloud storage space.

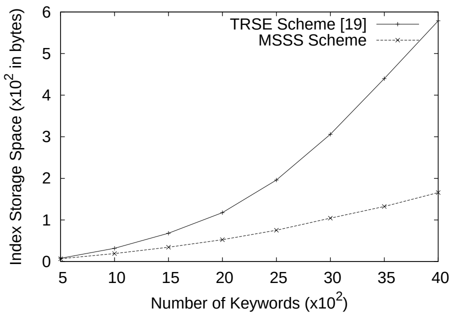

Fig. 3. Comparison of Index Storage Space based on Number of Keywords.

TABLE III
QUERIED KEYWORDS SEARCH TIME BASED ON NUMBER OF FILES

No. of Files	Queried Keywords Search Time (in s)	
	Abstract	
	TRSE [19]	MSMS
200	3.910	0.389
400	10.538	0.865
500	16.592	1.130
600	19.855	1.398
800	31.123	1.844
1000	45.083	2.485

VI. PERFORMANCE

We evaluate the overall performance of our proposed scheme on the real data set: National Science Foundation Research Awards Abstracts 1990-2003[23]. Our experiment environment involves a server and a client. The entire system is implemented in JAVA language on a Linux platform with Intel Xeon(R) CPU E31220 @3.10GHz Quad Core processor. The index file I'' and the encrypted collection of files F' is stored on the commercial public cloud, Amazon cloud services like $S3$ (simple storage service).

A. Performance on Storage Cost

Figure 3 shows the graph of index storage space of TRSE and MSSS scheme. The storage overhead is more in TRSE scheme, which does not support large scale data set. The space complexity of the TRSE is $O(D \times T)$, here terms are taken as maximum unique words from n files. If the term T is taken as 500 keywords and D is taken as 5 files. The total storage space taken by TRSE is $D \times T$ i.e., 5×500 = 2500 elements $\approx$ 7.8 KB whereas MSSS scheme takes $O(N_T \times C)$ storage space. N_T =623 and C = 3 for 5 files, $N_T \times C$= 623×3 = 1869 elements $\approx$ 6.8 KB is the storage space requried for the same set of files. From numerical analysis, we observe that MSSS scheme uses less storage space than TRSE scheme. The difference in storage increases exponentially with the increase in number of keywords.

B. Keyword Search Computation Time

Table III shows queried keyword search time based on number of keywords and number of documents respectively. The time complexity of keyword search on encrypted cloud data is $O(D \times T)$ for TRSE scheme and $O(B)$ for MSSS scheme where, B is number of terms present in the bucket out of total number of terms. The MSSS has three different cases 1) Best case is $O(1)$, if the bucket has only one term. 2) Average case is $O(B)$, if the bucket has B terms out of $N_T(1 < B < N_T)$. 3) Worst case is $O(N_T)$, if the bucket has all the rows in the index. Example: if D = 1,000 files and T = 20,387 terms, $D \times T$ = 1,000$\times$20,387 $\approx$ 20 million elements and search time is 45083 seconds for queried keyword $Abstract$ to retrieve top-k list of file ids in TRSE scheme. Similarly, if there are 1,000 files in MSSS scheme, N_T = 2,93,842 elements in the index file. If the Search query is $Abstract$, then it takes 2485 seconds to search on bucket A, which groups with the same starting letter A that contains 30,050 elements out of 2,93,842 elements. We observe that the search time of MSSS scheme is reduced by 94.4 $\%$ in comparison to TRSE scheme for keyword $Abstract$.The multi-keyword search time of MSMS scheme is reduced by 88 $\%$ than the TRSE scheme. Table III shows search time linear growth on different number of files over encrypted cloud storage data.

VII. Conclusions

In this paper, we have proposed a Most Significant Single-keyword Ranked Search over encrypted cloud data that supports efficient and accurate search. MSD radix sort algorithm is used to sort the keywords in the index file and counting sort algorithm is used to group the keywords with the same ASCII value into a bucket. The proposed MSSS scheme is more efficient than the existing TRSE scheme and also supports a large number of data files. MSSS scheme reduces computation and index storage overhead in comparison to earlier TRSE scheme. Finally, the proposed MSSS algorithm was performed on the real data set which shows reduction in index generation time, index storage space and keyword search time. Our Future work is to reduce search time and index storage space on multimedia.

References

[1] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, “Cloud Computing and Emerging IT Platforms: Vision, Hype, and Reality for Delivering Computing as the 5th Utility,” *Future Generation Computer Systems*, vol. 25, no. 6, pp. 599–616, 2009.

[2] D. X. Song, D. Wagner, and A. Perrig, “Practical Techniques for Searches on Encrypted Data,” *IEEE Symposium on Security and Privacy*, pp. 44–55, 2000.

[3] D. Boneh, G. Di Crescenzo, R. Ostrovsky, and G. Persiano, “Public Key Encryption with Keyword Search,” *in Proceedings of the Advances in Cryptology-Eurocrypt 2004*, pp. 506–522, 2004.

[4] A. Singhal, “Modern Information Retrieval: A Brief Overview,” *IEEE Data Eng. Bull.*, vol. 24, no. 4, pp. 35–43, 2001.

[5] C. Wang, N. Cao, J. Li, K. Ren, and W. Lou, “Secure Ranked Keyword Search over Encrypted Cloud Data,” *in Proceedings of the IEEE 30th International Conference on Distributed Computing Systems (ICDCS)*, pp. 253–262, 2010.

[6] 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. 222–233, 2014.

[7] X. Sun, X. Wang, Z. Xia, Z. Fu, and T. Li, “Dynamic Multi-Keyword Top-k Ranked Search over Encrypted Cloud Data.” *in International Journal of Security & Its Applications*, vol. 8, no. 1, pp. 319–332, 2014.

[8] C. L. Cheng, C. J. Sun, X. L. Xu, and D. Y. Zhang, “A Multi-Dimensional Index Structure based on Improved VA-file and CAN in the Cloud,” *in International Journal of Automation and Computing*, vol. 11, no. 1, pp. 109–117, 2014.

[9] C. Orencik, M. Kantarcioglu, and E. Savas, “A Practical and Secure Multi-Keyword Search Method over Encrypted Cloud Data,” *in Proceedings of the IEEE Sixth International Conference on Cloud Computing (CLOUD)*, pp. 390–397, 2013.

[10] X. Sun, Y. Zhu, Z. Xia, J. Wang, and L. Chen, “Secure Keyword-based Ranked Semantic Search over Encrypted Cloud Data,” *in Proceedings of the Advanced Science and Technology Letters(MulGraB 2013)*, vol. 31, pp. 271–283, 2013.

[11] S. Zerr, D. Olmedilla, W. Nejdl, and W. Siberski, “Zerber+r: Top-k Retrieval from a Confidential Index,” *in Proceedings of the 12th International Conference on Extending Database Technology: Advances in Database Technology*, pp. 439–449, 2009.

[12] C. Wang, N. Cao, K. Ren, and W. Lou, “Enabling Secure and Efficient Ranked Keyword Search over Outsourced Cloud Data,” *in IEEE Transactions on Parallel and Distributed Systems*, vol. 23, no. 8, pp. 1467–1479, 2012.

[13] P. Xu, H. Jin, Q. Wu, and W. Wang, “Public-Key Encryption with Fuzzy Keyword Search: A Provably Secure Scheme under Keyword Guessing Attack,” *in IEEE Transactions on Computers*, vol. 62, no. 11, pp. 2266–2277, 2013.

[14] C. H. Wang and C.-C. Hsu, “Integration of Hierarchical Access Control and Keyword Search Encryption in Cloud Computing Environment,” *in International Journal of Computer and Communication Engineering*, vol. 3, no. 2, pp. 333–337, 2013.

[15] Q. Liu, G. Wang, and J. Wu, “An Efficient Privacy Preserving Keyword Search Scheme in Cloud Computing,” *in Proceedings of the International Conference on Computational Science and Engineering CSE’09*, vol. 2, pp. 715–720, 2009.

[16] S. KumarVerma, S. Mathew, S. Srivastava, and S. Venkataesan, “An Efficient Dictionary and Lingual Keyword based Secure Search Scheme in Cloud Storage,” *in International Journal of Computer Applications*, vol. 68, no. 15, pp. 40–43, 2013.

[17] Z. Jiang and L. Liu, “Secure Cloud Storage Service with an Efficient DOKS Protocol,” *in Proceedings of the IEEE International Conference on Services Computing (SCC)*, pp. 208–215, 2013.

[18] P. Golle, J. Staddon, and B. Waters, “Secure Conjunctive Keyword Search over Encrypted Data,” *in Proceedings of the Applied Cryptography and Network Security*, pp. 31–45, 2004.

[19] J. Yu, S. J. T. P. Lu, Y. Zhu, G. Xue, and M. Li, “Toward Secure Multikeyword Top-k Retrieval over Encrypted Cloud Data,” *in IEEE Transactions on Dependable and Secure Computing*, vol. 10, no. 4, pp. 239–250, 2013.

[20] J. Kärkkäinen and T. Rantala, “Engineering Radix Sort for Strings,” *String Processing and Information Retrieval*, vol. 5280, pp. 3–14, 2009.

[21] B. A. Wagar, “System for MSD Radix Sort Bin Storage Management,” *US Patent 5,440,734*, Aug 8 1995.

[22] S. Ruggieri, “Efficient c4. 5 [Classification Algorithm],” *IEEE Transactions on Knowledge and Data Engineering*, vol. 14, no. 2, pp. 438–444, 2002.

[23] “National Science Foundation Research Awards Abstracts 1990-2003,” *http://kdd.ics.uci.edu/databases/nsfabs/nsfawards.html*, 2013.