IJIRCCE

e-ISSN: 2320-9801 | p-ISSN: 2320-9798

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN COMPUTER & COMMUNICATION ENGINEERING

Volume 8, Issue 8, August 2020

INTERNATIONAL STANDARD SERIAL NUMBER INDIA

Impact Factor: 7.488

9940 572 462 | 6381 907 438 | ijircce@gmail.com | www.ijircce.com

International Journal of Innovative Research in Computer and Communication Engineering

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | Impact Factor: 7.488 |

|| Volume 8, Issue 8, August 2020 ||

Chaotic Multi-Keyword Searchable Encryption for Mobile Cloud Storage

Preetha A[1], Seenivasan D[2]

B.E Student, Department of Computer Science and Engineering, K.S. Rangasamy College of Technology, Tiruchengode, Tamilnadu, India[1]

Assistant Professor, Department of Computer Science and Engineering, K.S. Rangasamy College of Technology, Tiruchengode, Tamilnadu, India[2]

ABSTRACT: Large amount of data can be stored in the cloud. To preserve the secrecy of the data and to retrieve the specific data, the data must be encrypted. Encrypted techniques play a major role when data are uploaded to the cloud, so that only the authenticated users can access. But searching the appropriate files in the cloud play a difficult task. To make this task in a easier way, provide a technique using Order Preserving Encryption (OPE). Cloud storage provides a easy accessible and massive storage at low cost, but data privacy is a major concern that prevents users from storing files on the cloud trustingly. One way of providing privacy from data owner point of view is to encrypt the files before uploading them onto the cloud and decrypt the files after downloading them.

KEYWORDS: cloud, order preserving encryption, authenticated, encrypted.

I.INTRODUCTION

Cloud computing in an unadulterated fitting and play model that drastically rearranges foundation arranging is the guarantee of „cloud computing". The two key preferences of this model are usability and cost-viability. Despite the fact that there remain inquiries on viewpoints, for example, security and merchant lock-in, the advantages this model offers are many. This work investigates a portion of the nuts and bolts of distributed computing with the point of presenting viewpoints, for example, Realities and dangers of the model Components in the model Characteristics and Usage of the model.

This work expects to give a methods for understanding the model and investigating choices accessible for supplementing your innovation and foundation needs. Appropriated processing is a figuring perspective, where a colossal pool of structures are related in private or open frameworks, to give continuously flexible establishment to application, data and record amassing. With the appearance of this innovation, the expense of calculation, application facilitating, content stockpiling and conveyance is diminished altogether.

Distributed computing is a down to earth way to deal with experience direct money saving advantages and it can possibly change a server farm from a capital-concentrated set up to a variable estimated condition. Cloud registering depends on an extremely major head of reusability of IT abilities. The distinction that distributed computing brings contrasted with customary ideas of "network registering", "disseminated figuring", "utility processing", or "autonomic processing" is to widen skylines across authoritative limits.

In an inexorably associated world, clients get to individual or shared information, put away "in the - cloud" with various gadgets. In spite of the notoriety of distributed storage administrations, little work has concentrated on researching distributed storage clients' Quality of Experience specifically on cell phones. Also, it isn't clear how clients' setting may influence QoE. Here led an online overview with 349 cloud administration clients to pick up understanding into their utilization and affordances.

To enrich the search functionality, the schemes supporting conjunctive keywords search have been proposed. Many works which supported the conjunctive keyword search, subset search, range queries were using the asymmetric encryption to achieve the conjunctive keywords search over encrypted data. In logarithmic-time search scheme was presented to support the range queries. Cao et al. proposed a privacy-preserving multi-keyword ranked search scheme using symmetric

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | Impact Factor: 7.488 |

|| Volume 8, Issue 8, August 2020 ||

encryption. Sun et al. proposed an efficient privacy-preserving multi-keyword supporting cosine similarity measurement. However, none of the schemes can support fuzzy keyword search.

Research efforts have been made to address the privacy-preserving fuzzy search problem over encrypted data. The existing solutions avoid the distance computation problem by enumerating all possible misspells as keywords. The expansion of the dictionary will clearly increase the storage overhead and eventually slows down the search computation effciency. Although fully homomorphic encryption is another solution to the problem, the computation is too expensive to scale for today's data volume.

II.EXISTING SYSTEM

2.1 Secure inverted index and search result verification

In the existing system the inverted index is an index data structure that stores the keyword-document mapping information. Different from the forward index which stores lists of words per document, the inverted index consists of document lists for each keyword in the dictionary. Each document list contains the IDs of all the documents that have the keyword. Compared with sequential iteration through each document for each keyword using the forward index, the inverted index limits the search within a subset of the documents. The inverted index is considered as the central component of a typical search engine indexing algorithm.

2.2 Multiple keywords searchable encryption with fuzzy search

Boldyreva and Chenette generalized Li et al.'s scheme to a primitive called efficiently fuzzy-searchable encryption (EFSE) for fuzzy search on encrypted data. The primitive works on a general closeness functions which translate data to a collection of tags. The tags of data represent the closeness which is the supported fuzziness by design. The authors also suggest using LSH functions as tag-encoding functions. Although this is a concurrent work, our scheme has two major contributions compared to theirs. First of all, our approach which utilizes Bloom filter is different from their tag-based scheme.

As a result, we eliminate the per-defined dictionary which is necessary for their scheme. Secondly, our scheme supports conjunctive keyword search. Although the EFSE can be extended to support connective search in multiple attribute such as in a database, it is infact achieved by combining multiple single attribute search results. Our scheme achieves the conjunctive keyword search through our innovative design and is computationally independent with the number of keywords in the query. Socek method is better than Grp and Cross methods and the chaotic maps performance remain similar even if an error rate slightly higher is observed with the original PWLCM. Statistical analysis on large amounts of images shows that on average, 8 to 16 adjacent pixels are correlated.

III.PROPOSED SYSTEM

CEES implementation for security enhancement for mobile cloud

In order to achieve security enhancement with energy and traffic efficiency, we implement the modules in CEES using modified routines and new algorithms. Our system will be introduced in three parts. So as to control the statistics information leak, we implement our one-to-many OPE in the data owner module.

CEES also wrap the keywords to be searched by adding some noise in the data user module to help controlling the keywords-files association leak. In order to get top-k relevant files, we implement a ranking function to calculate the relevant score on the cloud. Given a keyword in ORS, the cloud server is in charge of calculating the relevance scores for the data user to get the corresponding top-k relevant files. Therefore, we implement both the unwrap and rank functions in the cloud server module. Hence these modules are modified compared with the traditional ones.

IV.MODULE DESCRYPTION

4.1 REDESIGN OF DATAOWNER MODULE

The way of building the index to support the ORS scheme by our one-to-many OPE and implement it to control the statistics information leak. The authentication between the data owner and the data user is also redesigned in order to ensure the security of CEES. We now elaborate the implementation of the index construction, the encryption functions and detail the authentication process.

TF-IDF: TF-IDF is the product of two statistics, term frequency and inverse document frequency. Various ways for determining the exact values of both statistics exist. In the case of the **term frequency (TF)** *t f (t , d),* the simplest choice is to use the raw frequency of a term in
a document, i.e. the number of times that term t occurs in document *d.*

4.2 BUILD INDEX

In CEES, the data owner starts by collecting the files he wants to store into the cloud. Consider a file set $\mathcal{F} = (F_1, F_2, \ldots, F_n)$ containing the number of $|\mathcal{F}|$ files, in which a term set $\mathcal{T} = (t_1, t_2, \ldots, t_m)$ and the number of $|\mathcal{T}|$ terms appear. We create a table of size $|\mathcal{F}| \times |\mathcal{T}|$
for all the files and all the terms, where the value at the i_{th} row and j_{th} column denotes the number of occurrences of the i_{th} term in the j_{th} file. This numerical value is the TF value. Then a constant *S* is chosen as a cofactor to standardize these occurrences with the size of the files.

Let *GenKey()*be the function that generates the keys. Let $\pi()$ be a hash function that encrypts the terms. In CEES, $\pi()$ is instantiated by a hash function such as MD5. When building an index, it executes following two steps:

1) First, the data owner starts by calling *GenKey()*, to generate a key α to encrypt the terms, a key β to encrypt the index, and the noise $\lambda > 0$; $\mu > 0$ to wrap the keywords. He then outputs $K = \{\alpha, \beta\}$ and $N = \{\lambda, \mu\}$.
2) Second, the data owner builds a secure index by calling *BuildIndex*$(\mathcal{K}, \mathcal{F})$(encrypted $\mathcal{N}$ is sent to cloud server) as described in Algorithm 1:

Algorithm 1 BuildIndex

Input:$\mathcal{K}, \mathcal{F}$

Output:$\mathcal{I}$

1. Extract the terms $T = (t_1, t_2, \ldots t_m)$ from the file set $\mathcal{F}$.

2. **For** $t_i \epsilon\ T$ do

3. Get the encrypted term $\pi_\alpha(t_i)$ and hash it to get its entry $\psi(\pi_\alpha(t_i))$ in the TF table.

4. **end for**

5. For $t_i\ \epsilon\ T$ and $1 \leq j\ \leq |\mathcal{F}|$ do

6. Calculate the term frequency tf_{ij} and get $\widetilde{tf_{ij}} = |(\frac{S}{|F_i|}) \times tf_{ij}|$.

7. **end for**

8. Compute $\varepsilon_\beta(tf_{ij})$, and store it in the index *I*.

9. return *I*;

4.3 ENCRYPT FUNCTION

An aforementioned OPE approach employing a one-to-one mapping can be used to $\varepsilon_\beta()$.

Algorithm 2 Key Generation

Input: *TF Table*

Output: $\tilde{G}(TF), \tilde{H}(TF)$

1. Get the distribution histogram ψ of the TF table and get TF_x as all TF values occur in ψ
2. **for** $tf_i \epsilon\ TF_x$ **do**
3. Get the occurrence c_i.
4. **end for**
5. Get $C = \sum_{i=1}^{|TF_x|} c_i$.
6. **for** $tf_i \epsilon TF_x$ **do**
7. Calculate $p_i = c_i/C$
8. **end for**
9. **for** $tf_i \epsilon\ TF_x$ **do**
10. **if** i==1 **then**
11. Get $G(tf_i) = 1$ and $H(tf_i) = floor\ (2^B \times p_i)$.
12. **else**
13. Get
$G(tf_i) = H(tf_{i-1}) + 1\ and\ H(tf_i) =$
$H(tf_{i-1}) + floor\ (2^B \times p_i)$.
14. **end if**
15. **end for**
16. **return** $\tilde{G}(TF), \tilde{H}(TF)$.

And if desiring to update our files or index with our mobile device, this energy efficiency algorithm will become a good choice for data owners. In CEES, there is only one round-trip communication for each keyword search, and the selected index is not transferred between the cloud and the user as depicted in the TRS case. This attribute significantly reduces any possible communication overload.

Algorithm 3 Order Preserving Encryption

Input: *t f*

Output: $E(tf)$

1: **for** $t_i \epsilon T$ *and* $1 \le j \le |\mathcal{F}|$ **do**
2: Get
$E(tf_{ij}), E(tf_{ij}) \xleftarrow{R} \{G(tf_{ij}), G(tf_{ij}) +$
$1, \dots, H(tf_{ij})\}$.
3: **end for**
4: return $E(tf)$.

4.4 EXPERIMENTAL ENVIRONMENT

In our experiments, use a data set of 1000 files with different sizes and a VM in the cloud with Dual vCPUs at 2.27GHz. An android smart phone with a CPU at 1GHz sends the queries as the mobile client of CEES through an about 8M wireless network. As energy consumption is critical for mobile devices, we evaluate CEES energy efficiency in this subsection.

We use a phone power monitor to accurately measure the system energy consumption. Although slight changes depending upon the environment might occur, the comparison is accurte as controlled trials were performed. Observe that

the energy consumption is reduced from 0.08mAh to 0.036mAh when searching and retrieving files of size 100KB, which means that ORS saves 55% energy compared to TRS. When searching and retrieving files of 1MB size, the energy consumption is reduced from 0.164mAh to 0.106mAh, that means a 35% energy saving. So, CEES provides a very efficient power consumption. For example, to exhaust our 1650mAh battery, ORS (of CEES) can perform 22000 retrievals while TRS could only retrieve 13000 files of size 600KB.

File Search and Retrieval Time

The test of FSRT for different files with size ranging from 100KB to 1MB. Then observe that the FSRT of PTS is the shortest since it does not have to perform any security computation. The FSRT of ORS is effectively reduced when compared to the one of TRS. This difference is due to the advantages of the CEES design in terms of relevance score calculation offloading, and thus leads to reduction of file search and retrieval process. The FSRT value of ORS is very near to the one of PTS, implying a very low cost to security on the mobile device. For example, CEES saves FSRT by 46% compared to TRS for files of size 100KB, and by 23% for 1MB files .

The file retrieval time only depends on the file size and network bandwidth. When offered a greater bandwidth, CEES becomes more efficient since downloading time of files becomes a bottleneck of other schemes. The decryption time of the files is equal in all schemes and it is therefore pointless to
measure it. The efficient FSRT of CEES is achieved by improving the process efficiency, since only a single round of communication and relevance score calculation offload are used. The searching process is analysed in Table 1.

TABLE 1
FSRT analyse of PTS, TRS and ORS

	PTS	TRS	ORS
Request/Response	190ms	370ms	190ms
Stemming and Encryption	0	10ms	10ms
Hash and Wrap	0	145ms	150ms
Server file search	80ms	70ms	75ms
Client file search	0	260ms	0
Sum	270ms	855ms	425ms

Without any security service, PTS (Plain Text search) does not spend any time on stemming and encryption; neither does it on hash and wrap. On the other hand, ORS and TRS provide encrypted search schemes with related overhead. As shown in Table 1, ORS can improve the “request/response” time significantly than TRS from 370ms to 190ms (saving 180ms), and eliminate the “client file search” time by offloading it onto the server (saving 260ms).

V. RESULTS AND DISCUSSION

In this work, we developed a new architecture, CEES as an initial attempt to create a traffic and energy efficient encrypted keyword search tool over mobile cloud storages. We started with the introduction of a basic scheme that we compared to previous encrypted search tools for cloud computing and showed their inefficiency in a mobile cloud context. Then we developed an efficient implementation to achieve an encrypted search in a mobile cloud. The security study of CEES showed that it is secure enough for mobile cloud computing, while a series of experiments highlighted its efficiency.

CEES is slightly more time and energy consuming than keyword search over plain-text, but at the same time it saves significant energy compared to traditional strategies featuring a similar security level. Based on CEES, this work can be extended to more other novel implementations. We have proposed a multi-keyword search scheme to make encrypted data search efficient. However, there are still some possible extensions of our current work remaining. The proposed algorithm supports the search with multiple keyword search that should be in the encrypted form. However, there are still some possible extensions of the current work remaining. As OPE algorithm is a simple one, another extension is to find a powerful algorithm which will not harm the efficiency. It should be extended using any of the new simpler methods other than the SE methods. As our OPE algorithm is a simple one, another extension is to find a powerful algorithm which will not harm the efficiency.

VI.CONCLUSION

The search proposed the first chaos based searchable encryption approach which also allows both ranked and multi-keyword searches on the encrypted data stored in the cloud. This approach guarantees the privacy and confidentiality of the user and the cloud provider who is semi-trusted in our case. The proposed method is designed to achieve effective retrieval of remotely stored encrypted data for mobile cloud computing scenarios.

This scheme is implemented and evaluated using two databases: RFCs and the Enron database. An Android program receives the user's input and encrypts it before getting the hash value and then wrap it into a tuple which is sent to the mobile cloud server.

REFERENCES

[1] B.Yang, X. Pang, Q. Du, and D. Xie, "Effective error-tolerant keyword search for secure cloud computing," J. Comput. Sci. Technol., vol. 29, no. 1, pp. 81–89, Jan. 2014

[2] D. Boneh and G. D. Crescenzo, "Public key encryption with keyword search," in Advances in Cryptology, vol. 3027, C. Cachin and J. Camenisch, Eds., New York, NY, USA: Springer, 2004, pp.506–522.

[3] S. Kamara, and K. Lauter, "Cryptographic cloud storage," in Proc. Financial Cryptography Data Security, 2010, pp. 136–149.

[4] S. Kamara, C. Papamanthou, and T. Roeder, "CS2: A searchable cryptographic cloud storage system," Microsoft Res.earch, Redmond, WA, USA, Tech. Rep.ort MSR-TR-2011-58, 2011.

[5] Y. Earn, R. Alsaqour, M. Abdelhaq, and T. Abdullah, "Searchable symmetric encryption: Review and evaluation," J. Theoretical Appl. Inf. Technol., vol. 30, p. 48, 2011.

[6] R. Koletka and A. Hutchison, "An architecture for secure searchable cloud storage," in Proc. IEEE Inf. Security South Africa, Aug. 2011, pp. 1–7.

[7] E. Stefanov, C. Papamanthou, and E. Shi, "Practical dynamic searchable symmetric encryption with small leakage," IACR Cryptology ePrint Archive, 2013.

[8] Na Chen, Lu Wang, Limin Jia, Honghui Dong, Haijian Li, "Parking Survey Made Efficient in Intelligent Parking Systems", Procedia Engineering, volume 137, pp. 487-495, ISSN 1877-7058, 2016.

[9] J. Bringer, H. Chabanne, and B. Kindarji, "Error-tolerant searchable encryption," presented at the IEEE Int. Conf. Communication, Dresden, Germany, Jun. 2009.

[10] J. Yu, J. Li, X. Wang, and W. Gao, "Conjunctive fuzzy keyword search over encrypted data in cloud computing," TELKOMNIKA Indonesian J. Elect. Eng., vol. 12, no. 3, pp. 2104–2109, Mar. 2014.

[11] C. Wang, N. Cao, J. Li, K. Ren, and W. Lou, "Secure ranked keyword search over encrypted cloud data," in Proc. IEEE 30th Int. Conf. Distrib. Comput. Syst., 2010, pp. 253–262

[12] R. Li, Z. Xu, W. Kang, K. Choong Yow, and C. Z. Xu, "Efficient multi-keyword ranked query over encrypted data in cloud computing," Future Gener. Comput. Syst., vol.30,pp.179–190,2014.

[13] J. Bringer, H. Chabanne, and B. Kindarji, "Identification with encrypted biometric data, "Security Commun. Netw., vol. 4, no. 5, pp. 548–562, May 2011

[14] M. Kuzu, M. S. Islm, and M. Kantarcioglu, "Efficient similarity search over encrypted data," in Proc. IEEE 28th Int. Conf. Data Eng., 2012, pp. 1156–1167

[15] A. Awad and A. Saadane, "Efficient chaotic permutations for image encryption algorithms," presented at the World Congr. Eng., vol. 1, London, U.K., 2010

[16]

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | Impact Factor: 7.488 |

|| Volume 8, Issue 8, August 2020 ||

[17] B. Julien, H. Chabanne, and B. Kindarji, "Identification with encrypted biometric data," Security Commun. Netw., vol. 4, no. 5, pp. 548–562, 2011

[18] S. Talari, M. Shafie-khah, P. Siano, V. Loia, A. Tommasetti, J. P. S. Catalão, "2017", "A Review of Smart Cities Based on the Internet of Things Concept", Energies, 10, 421, pp:1-23,.

[19] M. Ismail, G. Chalhoub, and B. Bakhache, (2012), "Evaluation of a fast symmetric cryptographic algorithm based on the chaos theory for wireless sensor networks," in Proc. IEEE 11th Int. Conf. Trust, Security Privacy Comput. Commun., pp. 913–919

[20] B. Julien, and B. Kindarji, (2011), "Identification with encrypted biometric data," Security Commun. Netw., Vol. 3, No. 5, pp. 548–562

INNO SPACE

SJIF Scientific Journal Impact Factor

Impact Factor:
7.488

ISSN INTERNATIONAL STANDARD SERIAL NUMBER INDIA

निस्केयर NISCAIR

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN COMPUTER & COMMUNICATION ENGINEERING

9940 572 462 6381 907 438 ijircce@gmail.com

www.ijircce.com

Scan to save the contact details