Siri Y S, 2021, 9:3
ISSN (Online): 2348-4098
ISSN (Print): 2395-4752

An Efficient Search Scheme Over Encrypted Data on Cloud

Siri Y S, Mandakini, Nikitha S, Prof. Shilpa Shree S.
Department of Information Science And Engineering,
Dayananda Sagar College of Engineering,
Bangalore-560078, Karnataka,
sirisumukh@gmail.com, shetteppalpote@gmail.com, nikithas163@gmail.com, shilpashree- ise@dayanandasagar.edu

Abstract- Cloud storage provides a convenient, massive, and scalable storage at low cost, but data privacy is a major concern that prevents users from storing files on the cloud trustingly. One way of enhancing privacy from data owner point of view is to encrypt the files before outsourcing them onto the cloud and decrypt the files after downloading them. However, data encryption is a heavy overhead for the mobile devices, and data retrieval process incurs a complicated communication between the data user and cloud. Normally with limited bandwidth capacity and limited battery life, these issues introduce heavy overhead to computing and communication as well as a higher power consumption for mobile device users, which makes the encrypted search over mobile cloud very challenging. In this paper, we propose traffic and energy saving encrypted search (TEES), a bandwidth and energy efficient encrypted search architecture over mobile cloud. The proposed architecture offloads the computation from mobile devices to the cloud, and we further optimize the communication between the mobile clients and the cloud. It is demonstrated that the data privacy does not degrade when the performance enhancement methods are applied. Our experiments show that TEES reduces the computation time by 23 to 46 percent and save the energy consumption by 35 to 55 percent per file retrieval; meanwhile the network traffics during the file retrievals are also significantly reduced.

Keywords:- Mobile Cloud Storage, Searchable Data Encryption, Energy Efficiency, Traffic Efficiency.

I. INTRODUCTION

Cloud storage system is a service model in which data are maintained, managed and backed up remotely on the cloud side, and meanwhile data keeps available to the users over a network. Mobile Cloud Storage (MCS) denotes a family of increasingly popular on-line services, and even acts as the primary file storage for the mobile devices.

MCS enables the mobile device users to store and retrieve files or data on the cloud through wireless communication, which improves the data availability and facilitates the file sharing process without draining the local mobile device resources.

The data privacy issue is paramount in cloud storage system, so the sensitive data is encrypted by the owner before outsourcing onto the cloud, and data users retrieve the interested data by encrypted search scheme.

In MCS, the modern mobile devices are confronted with many of the same security threats as PCs, and various traditional data encryption methods are imported in MCS. However, mobile cloud storage system incurs new challenges over the traditional encrypted search schemes, in consideration of the limited computing and battery capacities of mobile device, as well as data sharing and accessing approaches through wireless communication.

International Journal of Science, Engineering and Technology

An Open Access Journal

Therefore, a suitable and efficient encrypted search scheme is necessary for MCS.
Generally speaking, the mobile cloud storage is in great need of the bandwidth and energy efficiency for data encrypted search scheme, due to the limited battery life and payable traffic fee.

Therefore, we focus on the design of a mobile cloud scheme that is efficient in terms of both energy consumption and the network traffic, while keep meeting the data security requirements through wireless communication channels.

II. FILE RETRIEVAL IN CLOUD STORAGE

1. Traditional Encrypted Search over Cloud Data:
Traditional cloud storage system architecture and general procedures, which include:

- File/index encryption by the data owner,
- Outsourcing the data to the cloud storage,
- Encrypted data search/retrieval procedure of the data users in cloud computing.

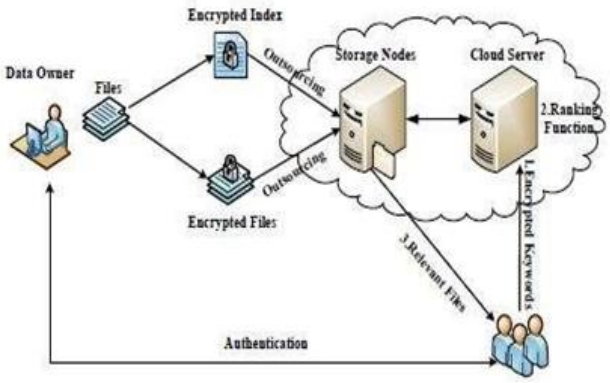

Fig 1. Traditional Encrypted Search Architecture.

1.1 File/Index Encryption: The data owner first executes the preprocessing and indexing work as shown below. He should invert files, that are selected to store on the cloud, for text search engines. Every word in these files undergoes stemming to retain the word stem. After this step, the data owner encrypts and hashes every term (word stem) to fix its entry in the index. The index is then created by the data owner.

Finally, the data owner encrypts the index and stores it into the cloud server, together with the encrypted file set. Most of the previous schemes under this architecture use Order Preserving Encryption (OPE) to encrypt the file index. This file index is often a TF (Term Frequency) table composed of TF values. The TF-IDF table could be used to determine word relevance in documents.

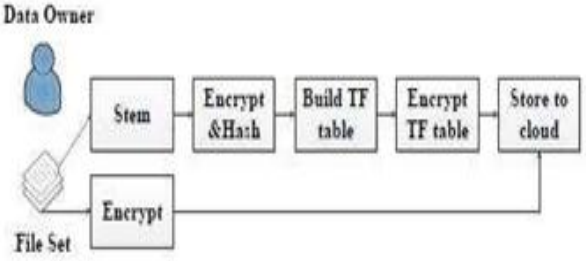

Fig 2. Process of Preprocessing and Indexing.

1.2 Data Search and Retrieval after Authentication:
A data user can only access a file after being authenticated by the data owner. In the process of authentication, the data user sends his identity to the data owner. The data owner sends the encrypted keys back if the user is a legal user. In the process of search and retrieval, the cloud server helps the users to find the top-k relevant files for a given keyword without decrypting it.

Searches incur following the steps, as illustrated:

- An authenticated user stems the keyword to be queried, encrypts it with the keys and hashes it to get its entry in the index. Then the encrypted keyword is sent to the cloud server.
- On receiving the encrypted keyword, the cloud server first searches for it in the index. Then the index related to this keyword is sent back to the data user.
- The data user calculates the relevance scores with the selected index to find the top-k relevant files and sends a follow-up request to the cloud server in order to retrieve the files.
- The position of these files is selected and they are sent back to the data user from the cloud server.
- The data user decrypts the files and recovers the original data.

The related computational components for these steps are illustrated in Figure 3, which indicate the traditional two-round-trip scheme for a file search and retrieval process invoked by an authenticated user.We call this file retrieval scheme abbreviated as TRS (Two Roundtrip Search).

This scheme provides privacy protection through a complicated file retrieval process compared to a simple PlainText Search scheme (PTS) where

International Journal of Science, Engineering and Technology

An Open Access Journal

searching and retrieving a file is done in only one round without security service.

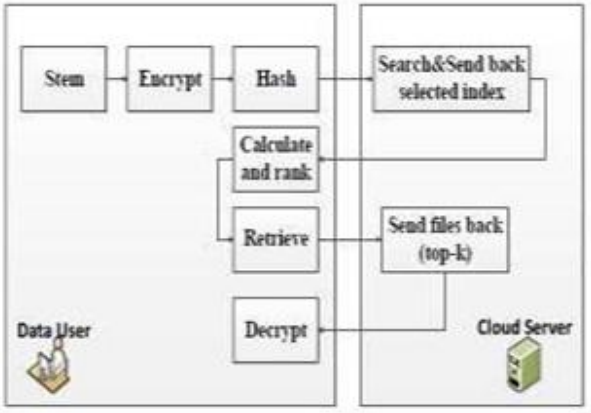

Fig 3. TRS: Two-Round-Trip Encrypted Search.

2. MCS Challenges and Design Principle:

2.1 Efficiency Challenges: The traditional file search and retrieval schemes, such as TRS, can provide data security but at the cost of more complicated procedures than Plain Text Search (PTS).

TRS has been widely employed in cloud storage systems, but the encryption and the ranking incur the heavy calculation cost on a mobile device, and thus introduce the new challenges inefficiency for MCS traffic and energy consumption. It is necessary to rethink the design of the whole procedure with a careful consideration of the energy consumption and of the traffic efficiency.

We analyse the model and indicate several possible optimizations. First, it is obvious that in traditional schemes, the mobile client has a heavy workload for decrypting the selected index, calculating and ranking the relevance scores. It will take more time when comes to the mobile client since the computing capacity of a mobile device is limited. This is also clearly inappropriate when the battery of a mobile device is taken into account. Second, the two round-trips for each file search and retrieval request, as shown in Figure 3, is a heavy burden for mobile devices with limited bandwidth and traffic fees.

In a bad network environment, more latency will be introduced in two round-trips than in one round-trip. Thus, we offload the calculation of the relevance scores to the cloud server; this reduces the burden on the mobile clients while also shortening the retrieval process.

As a result, the data user can receive the most relevant files within only one communication.

2.2 Security Challenges: According to the efficiency challenges in cloud storage mentioned before, we should then address the security challenges introduced by offloading part of the calculation onto the cloud. We consider the scenario where an authorized data user wants to search for files stored on the cloud server. This data user needs to retrieve the most relevant files through the encrypted data without downloading all the files.

So the index should be stored in the cloud, leading to potential threats for MCS in the following cases:

2.2.1 Statistics Information Leak. Attackers could get the terms by analyzing the TF table, since an Order Preserving Encryption (OPE) method encrypted TF table produces a peaky histogram of TF values. In other words, term frequency should be evenly distributed to avoid statistic information leak, otherwise a broken index can be introduced with serious information leaks.

2.2.2 Keywords-files Association Leak. An attacker could determine query terms by observing queries and results through a wireless channel: as the result of the retrieval is keyword specific, attackers may guess the queried keyword by only observing the keyword and the result of the retrieval. Thus, it should avoid the relation of this keywords-files association in data encryption,

2.2.3 Server Information Acquisition. The cloud server maybe honest-but-curious , and my try to learn the underlying plaintext of users data. The cloud server can infer and analyze the encrypted index and get additional information, and we need to minimize the information acquisition of the curious cloud server.

3. Design Principle:

Our design goal is to achieve an efficient encrypted search architecture, while both considering these security threats in the modified implementation. Our scheme will offload most of the computational load to the cloud server as described.

Nextly, presents the details of its implementation with security enhancement. Note that the novel TEES architecture design must base on an encrypted search method, which traditionally include single-keyword search and multiple keyword search.

International Journal of Science, Engineering and Technology

An Open Access Journal

Multiple-keyword search enables conjunctive or disjunctive search formulas, but it usually incurs high computational complexity to realize multi dimensional range query over encrypted data due to the heavy reliance on public-key cryptography. In consideration of the limited data volume for MCS users and limited computing capacity of mobile devices, single keyword search is more suitable. Therefore, we dedicate in the single-keyword search algorithm to provide secured and lightweight searchable encryption solution for mobile cloud storage and file sharing system design.

III. TEES SYSTEM DESIGN

To effectively support an encrypted search scheme with a high security level over cloud data, we introduce a new architecture that we name TEES, our aim is to design a practical solution for secure encrypted search over a mobile cloud storage and then introduce development of our own protocol with the change of the traditional process of file search and retrieval for the cloud data in our scheme achieves the security and efficiency goals mentioned above.

1. Modified Process of Search and Retrieval:

During the preprocessing and indexing stages, the data owner gets a TF table as index and uses Order Preserving Encryption (OPE) to encrypt it. As a result, the cloud server is able to calculate the relevance scores and rank them without decrypting the index. This renders the offloading of the computational load secure and possible.

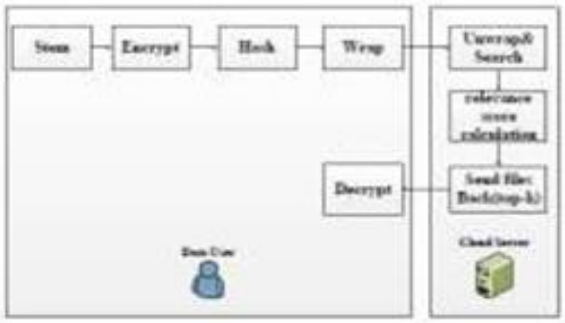

Fig 4. ORS: Novel Process of Search and Retrieval.

Thus, the modified search and retrieval process is:

- If a data user wants to retrieve the top-k relevant files based on a keyword, he first obtains authentication from the data owner and then receives the keys to encrypt the keyword.
- The data user stems the keyword to be queried and encrypts it using the keys.
- The data user wraps the encrypted keyword into a tuple, adding some noise to avoid statistic information leak; this tuple is used to perform the retrieval. Then, it is sent to the cloud server together with the number k. The wrap method renders the keywords indistinguishable for an attacker.
- On receiving the wrapped keyword, the cloud server first makes sure that it is accessed by a legal user. If the server is notified by the data owner that this user is to become invalid in a near future, the search is performed but a warning is also issued. If this is a legal user, the server unwraps the tuple to recover the entry of the keyword and searches for it in the index. After calculating the relevance scores, the position of the files corresponding to the keyword is picked and the top k relevant files are sent back to the data user's mobile clients without performing any decryption on these files.
- The data user decrypts these files in the mobile client and recovers the original data.

IV. TEES IMPLEMENTATION: FOR SECURITY ENHANCEMENT, FOR MOBILE CLOUD

In order to achieve security enhancement with energy and traffic efficiency, we implement the modules in TEES using modified routines and new algorithms. Our system will be introduced in three parts. As previously mentioned, the data owner should build a TF table as index and encrypt it using OPE in order to offload the calculation and ranking load of the relevance scores to the cloud. So as to control the statistics information leak, we implement our one-to-many OPE in the data owner module. We 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

International Journal of Science, Engineering and Technology

An Open Access Journal

modules are modified compared with the traditional ones.

1. Redesign of the Data Owner Module:

We modify 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 TEES.

We now elaborate the implementation of the index construction, the encryption functions and detail the authentication process.

1.1 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) tf (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. If we denote the raw frequency of t by f(t; d), then the simple TF scheme is tf (t; d) = f(t; d).

The inverse document frequency (IDF) is a measure of whether the term t is common or rare across all documents. It is obtained by dividing the total number of documents by the number of documents containing the term, and then taking the logarithm of that quotient.

Then TF-IDF is calculated as:tf{idf(t; d;D) = tf (t; d) _ idf(t;D); (4) where D denotes the total number of documents in the data set. A high weight in TF-IDF is reached by a high term frequency (in the given document) and a low document frequency of the term in the whole collection of documents; the weights thus tend to filter out common terms. Since the ratio inside the IDF's log function is always greater than or equal to 1, the value of IDF (and TF-IDF) is greater than or equal to 0. As a term appears in more documents, the ratio inside the alogarithm approaches 1, bringing the IDF and TF-IDF closer to 0.

1.2 Build Index: In TEES, the data owner starts by collecting the files he wants to store into the cloud. Consider a file set F= (F1; F2; :::; Fn) containing the number of jFj files, in which a term set T = (t1; t2; :::; tm) and the number of jT j terms appear. We create a table of size jFj_jT j for all the files and all the terms, where the value at the ith row and jth column denotes the number of occurrences of the ith term in the jth 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. In TEES, we use this TF table as our index and the cloud server calculates the relevance scores using the encrypted TF values. The ranking function of the relevance scores will be introduced in the subsection dedicated to the cloud server.

1.3 Encrypt Function: An aforementioned OPE approach employing a one-to-one mapping can be used. However such an OPE strategy cannot be directly employed to secure the TF values in consideration of security issues in MCS. In order to flatten the distribution, we use a one-to-many OPE, i.e., we map every TF value to a random number in a certain range. For example, a TF value tf will be mapped to a range [tfx; tf y], where 0 _ tfx _ tf y < 2B are the lower and upper bound of the random mapping range (B is set to 16 in our performance evaluation and to 8 in our security evaluation).For two adjacent TF values OPE flattens the TF values histogram over a file set so as to ensure the system security. OPE is a secure algorithm for encryption which is proved by Boldyreva et al., so that our OPE algorithm is secure enough for daily use.

Meanwhile, our algorithm is a simple implementation of order-preserving encryption which will consume less energy than other complicated ones. And if we are desiring to update our files or index with our mobile device, this energy efficiency algorithm will become a good choice for data owners. The data owner can also use AES (Advanced Encryption Standard) to encrypt the files.

1.4 Authentication: In TEES, the data owner maintains a set of legal users ("legal set") and a set of users that will become invalid in after a defined delay ("overdue set"). The process of authentication is shown on Figure below.

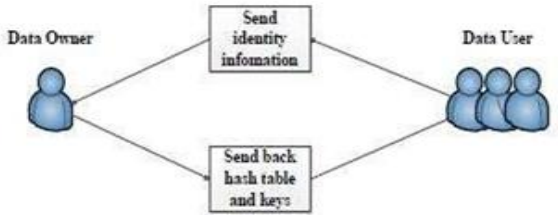

Fig 5. Process of Authentication.

When a user intends to access the file, he first sends his information to be authenticated by the data owner. In our design, we use our unified school authentication in TEES and transfer it through https for safety concern. The data owner sends the keys along with the hash table back if the user belongs to the legal set.

Then the data owner records the "International Mobile Equipment Identity" of the user's mobile device and stores its encrypted version into the cloud. When the user's authority is overdue, his identity information is moved to the "overdue" set. The data owner will also notify the cloud of the changes.

2. Redesign of the Data User Module:
The data user module is executed on the mobile clients side. The wrap function of the keywords is implemented to solve the keywords-files association leak. In the wrap function, the stem, the encryption and the hash operation are exactly the same as in the index building algorithm.The function decrypting the files corresponds to the encryption done by the data owner. The authentication function is used for authentication. We now detail the wrap function of this module.

2.1 Wrap Function with Noise: When an authorized data user wants to retrieve files, he needs to encrypt the corresponding query keyword w, and get the hash value h from the hash table. This hash value is then sent to the cloud server and used to compute the relevance scores. In order to render this hash value indistinguishable for an attacker, the cloud client should wrap it, adding some noise before sending it to the cloud server. The wrap function Wrap() will, first of all create a random number, and then build a tuple (h1; h2) .

2.2 Ranking Function: Cloud server calculates the relevance scores and return top-k relevant files according to the searching query from data user. The calculation scheme is used in our scheme. Note that due to the order preserving index, any other relevance scores calculation method can also be employed. We can also find the probability for a term to appear in a file using a middle tier "topic", and then store its encrypted value in the index.Moreover, it is known that multiple- keywords search can provide more accurate the search results. Overall, developing a single keyword search scheme is a proper solution of encrypted search data sharing for mobile cloud storage. Moreover, TEES is a general architecture, where the OPE method proposed here can be substituted by other novel schemes.

V. SECURITY ANALYSIS AND EVALUATION

The most important principle of the design is to prevent the attacker from obtaining any plaintext information regarding our data file set or the searched keyword. Then we should let the trusted but curious mobile cloud server learn as little information as possible.

Last but not least, an unauthenticated data users should not be able to perform any file retrieval. We ran experiments to test the security of TEES.

1. Statistics Information Leak Control:
TEES protects the terms from being determined by analyzing the distribution of the TF values through mobile cloud communication channels. Figure below shows the histogram of the TF values over a data set, and we can see that the TF histogram are very sharp originally.

It means that an attacker may get statistical informations from the TF table as previously explained. In TEES, we encrypt the TF table with one-to-many order preserving encryption. Every TF value is mapped to a mapping range. We indeed obtain an approximately uniform distribution once the order preserving encryption has been applied.

This means that even if an attacker accesses the TF table, it is hard to gain any extra information on our data set in a short time.

2. Keywords-files Association Leak Control:
TEES also enhances the security of the keywords by preventing the attacker from observing the keywords to be searched as well as the results of a search. Then this hash value is embedded into a tuple (h1; h2) and sent to the server.

Even if the attacker knows the content to be queried, so he is also unable to determine the terms. The tuple (h1; h2) corresponding to the word "Bund" of our data set in 200 retrievals is distributed after being wrapped.

International Journal of Science, Engineering and Technology

An Open Access Journal

VI. RUN TIME PERFORMANCE EVALUATION

In addition to the system security analysis and evaluation, we now evaluate TEES performance in terms of energy, traffic and file access. We will compare its performances to those of TRS and PTS schemes.

1. Experimental Environment:

We 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 TEES through an about 8M wireless network.

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. Another feature of this program is to retrieve the files back from the mobile cloud server and decrypt them. In addition, we implemented both TRS and PTS for a comparative purpose.

2. Energy Consumption:

As energy consumption is critical for mobile devices, we evaluate TEES energy efficiency in this subsection. The energy consumption of TRS and ORS is shown in Figure below.

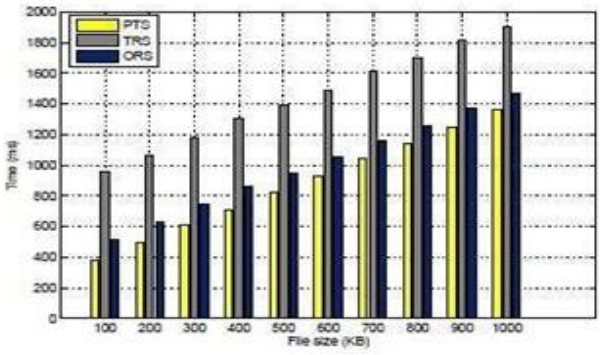

Fig 6.

Although slight changes depending upon the environment might occur, the comparison is quite accurate 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, TEES provides a very efficient power consumption. For example, to exhaust our 1650mAh battery, ORS (of TEES) can perform _22000 retrievals while TRS could only retrieve _13000 files of size 600KB.

VII. EQUATIONS USED IN TEES

1. Prediction of Energy Consumption Is Reduced Significantly:

$$\eta = \frac{E_{co} + E_{retr}}{2E_{co} + E_{retr} + E_w} < 100\%$$

2. Reducing File Search and Retrieval Time:

$$\rho = \frac{RTT + \frac{\Sigma}{C_{cs}} + T_{retr}}{2RTT + \frac{\Sigma}{C_m} + T_{retr}} < 100\%.$$

3. Reducing the Energy Consumption:

$$\zeta = \frac{C_{kw} + C_{rt} + C_f}{C_{kw} + 2C_{rt} + C_{pr} + C_f} < 100\%$$

4. Reducing Traffic Overhead:

$$tf\text{–}idf(t, d, D) = tf(t, d) \times idf(t, D)$$

VIII. STATISTICS GRAPH AND TABLE

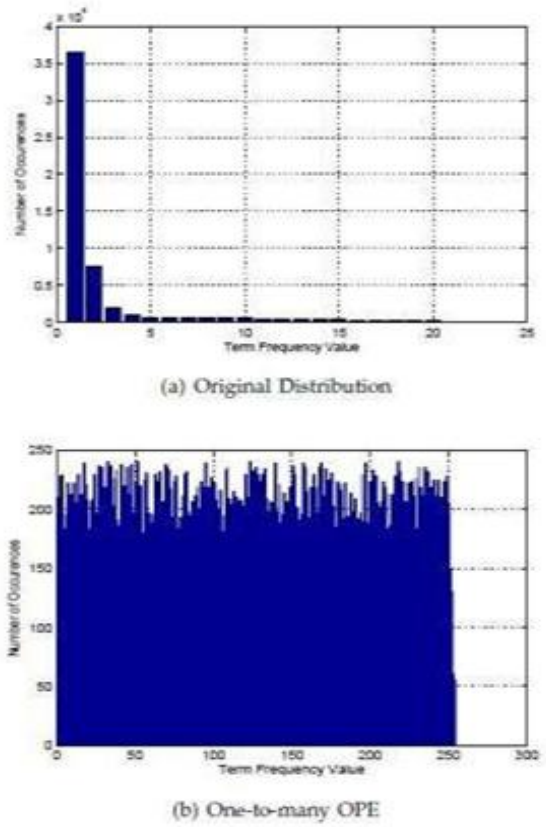

Fig 7. Graph 1.

International Journal of Science, Engineering and Technology

An Open Access Journal

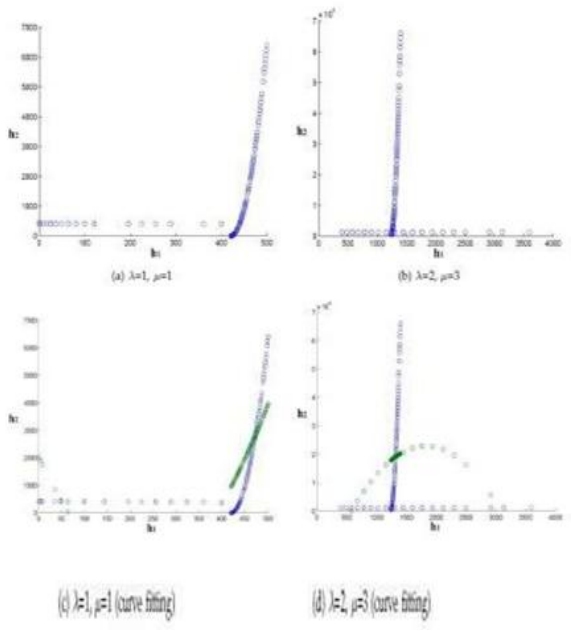

Fig 8. Graph 2.

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

IX. RELATED WORK

1. Encrypted Search Schemes:

Over the past recent years, encrypted search has evolved toward the ability data sharing with protection of users' privacies raised the question how to do keyword searches on encrypted data efficiently. They proposed a scheme which encrypted each word of a document separately.

So it is not compatible with existing file encryption schemes and it cannot deal with compressing data. After that many methods of keyword search showed up such as. In Information Retrieval, TF-IDF (term frequency-inverse document frequency) is a statistic which reflects how important a word is to a document in a collection or corpus. It is often used as a weighting factor in keyword-based retrieval and text mining. The TF-IDF algorithm proposed by Salton and McGill's book is one of the most popular schemes.

2. Power and Traffic Efficiency Improvements Schemes:

The previous schemes cannot directly apply to mobile cloud, for achieving efficient energy consumption to address the important issue for mobile cloud. In recent years many OPE or fully homomorphic encryption methods have been proposed.

They proved themselves secure and accurate enough for searching encrypted data purpose. However, they cost many computing resources. As energy consumption becoming important, a complicated algorithm is not suitable in mobile devices. Therefore we choose a simple order preserving encryption method in our TEES.

X. RESULTS

Fig 9. Home Page.

Fig 10. Data Provider Login Page.

International Journal of Science, Engineering and Technology

An Open Access Journal

Fig 11. Registration Page.

Fig 12. Login Page.

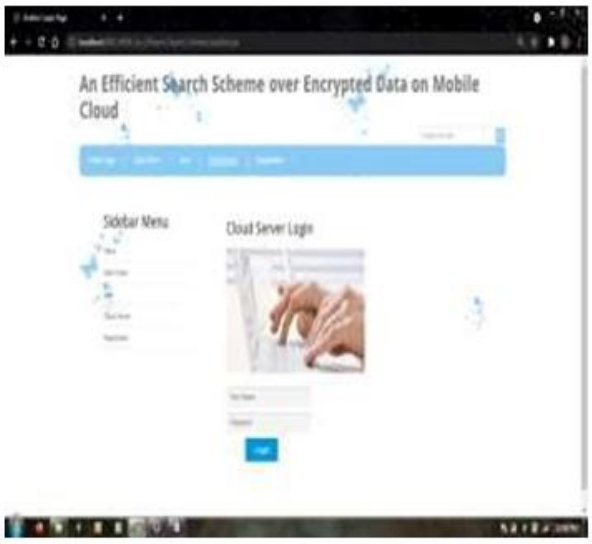

Fig 13. Cloud Server login Page.

Fig 14. Verification.

Fig 15. Search Single/Multiple Keyword.

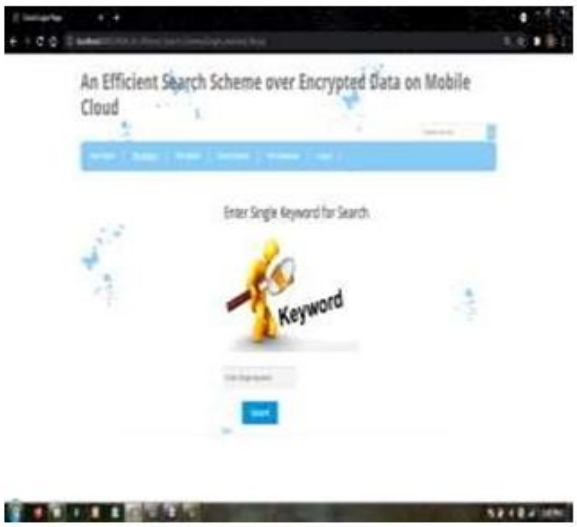

Fig 16. Single Search.

International Journal of Science, Engineering and Technology

An Open Access Journal

Fig 17. Access Found.

Fig 18. Multiple Keyword Search.

XI. RESEARCH METHODOLOGY

In this paper, TEES, which has more bandwidth and better energy efficient encrypted search over a mobile cloud. The proposed architecture removes the computation from mobile devices to
the cloud, and hence we further can optimize the communications of the mobile clients and the cloud.

As energy consumption is becoming necessary, a complex algorithm is not suitable in mobile devices. Hence, we choose a simple order preserving encryption method in our TEES.

Data privacy is a major concept that prevents users from storing files on the cloud. One way of improving the data privacy is to encrypt the files before sending them onto the cloud and decrypt the files when they are downloaded.

XII. CONCLUSION AND FUTURE WORK

In this paper, we developed a new architecture, TEES 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.

We have proposed a single keyword search scheme to make encrypted data search efficient. However, there are still some possible extensions of our current work remaining. We would like to propose a multi-keyword search scheme to perform encrypted data search over mobile cloud in future. As our OPE algorithm is a simple one, another extension is to find a powerful algorithm which will not harm the efficiency.

REFERENCES

[1] L. Vaquero, L. Rodero-Merino, J. Caceres, and M. Lindner, "A break in the clouds: towards a cloud definition," ACM SIGCOMM Computer Communication Review, vol. 39, no. 1, pp. 50–55, 2008.

[2] X. Yu and Q. Wen, "Design of security solution to mobile cloud storage," in Knowledge Discovery and Data Mining. Springer, 2012,pp. 255–263.

[3] D. Huang, "Mobile cloud computing," IEEE COMSOC Multimedia Communications Technical Committee (MMTC) E-Letter, 2011.

[4] O.Mazhelis, G. Fazekas, and P. Tyrvainen, "Impact of storage acquisition intervals on the cost-efficiency of the private vs. public storage," in Cloud Computing (CLOUD), 2012 IEEE 5th International Conference on. IEEE, 2012, pp. 646–653.

[5] J. Oberheide, K. Veeraraghavan, E. Cooke, J. Flinn, and F. Jahanian, "Virtualized in-cloud security services for mobile devices," in Proceedings of the First Workshop on Virtualization in Mobile Computing. ACM, 2008, pp. 31–35.

[6] J. Oberheide and F. Jahanian, "When mobile is harder than fixed (and vice versa): demystifying security challenges in mobile environments," in Proceedings of the Eleventh Workshop on Mobile Computing Systems & Applications. ACM, 2010, pp. 43–48.

International Journal of Science, Engineering and Technology

An Open Access Journal

[7] A. A. Moffat, T. C. Bell et al., Managing gigabytes: compressing and indexing documents and images. Morgan Kaufmann Pub, 1999.

[8] D. Song, D. Wagner, and A. Perrig, "Practical techniques for searches on encrypted data," in Security and Privacy, 2000. S&P 2000. Proceedings. 2000 IEEE Symposium on. IEEE, 2000, pp. 44–55.

[9] D. Boneh, G. Di Crescenzo, R. Ostrovsky, and G. Persiano, "Public key encryption with keyword search," in Advances in Cryptology- Eurocrypt 2004. Springer, 2004, pp. 506–522.

[10] R. Curtmola, J. Garay, S. Kamara, and R. Ostrovsky, "Searchable symmetric encryption: improved definitions and efficient constructions," in Proceedings of the 13th ACM conference on Computer and communications security. ACM, 2006, pp. 79–88.

[11] Y. Chang and M. Mitzenmacher, "Privacy preserving keyword searches on remote encrypted data," in Applied Cryptography and Network Security. Springer, 2005, pp. 391–421.

[12] S. Zerr, D. Olmedilla, W. Nejdl, and W. Siberski, "Zerber+ r: Topk retrieval from a confidential index," in Proceedings of the 12th International Conference on Extending Database Technology: Advances in Database Technology. ACM, 2009, pp. 439–449.