EnDAS: Efficient Encrypted Data Search as a Mobile Cloud Service

[1]V. Bhuvaneshwari, [2]B. Deeba, [3]N. Balasubramanian,[4]M. Mohamed Rafi

[1]Final year MCA, Mohamed Sathak Engineering College, Kilakarai

[2]Associate Professor, Dept of MCA, Mohamed Sathak Engineering College, Kilakarai

[3]Associate Professor, Dept of MCA, Mohamed Sathak Engineering College, Kilakarai

[4] Professors, Dept of MCA, Mohamed Sathak Engineering College, Kilakarai

Abstract—Document storage in the cloud infrastructure is hurriedly in advance popularity throughout the world. However, it poses risk to consumers unless the data is encrypted for protection. Encrypted data should be well searchable and retrievable without any privacy leak, mostly for the mobile client. Even though new research has solved many security issues, the architecture cannot be useful on mobile devices frankly under the mobile cloud surroundings. This is due to the challenges forced by wireless networks, such as latency sensitivity, poor connectivity, and low transmission rates. This leads to a long search time and extra network traffic costs when using traditional search schemes. This study addresses these issues by proposing an efficient Encrypted Data Search (EnDAS) scheme as a mobile cloud service. This innovative system uses a lightweight trapdoor compression method, which optimizes the information announcement progression by reducing the trapdoor's size for traffic efficiency. In this study, we also suggest two optimization methods for manuscript investigate, called the Trapdoor Mapping Table module and Ranked Serial Binary Search algorithm, to speed the search time. Results demonstrate that EnDAS reduces search time by 38% to 50% as well as network traffic by 12% to 45%.

Index Terms—Mapping board, Compression, Ranking explore, Encrypted Search, Mobile Cloud.

1. INTRODU1CTION

Cloud Computing is the exploit of hardware and software to carry a tune-up over a network. With cloud computing, users can access records and use applications from any device that is able to access the Internet. A paradigm of a Cloud Computing provider is Google's Gmail. Cloud computing is a type of computing that relies on shared computing resources rather than having local servers or personal strategy to handle applications. In its most trouble-free narrative, cloud computing is taking services and moving them exterior an organization's firewall. Applications, storage and other services are accessed via the Web. The services are delivered and used over the Internet and are paid for by the cloud customer on an as-needed or pay-per-use business model.

As cloud computing can shore up flexible services and provide an cost-effective use of storage and computation resources, it is fast in advance popularity. In the midst of powerful cloud services, many data providers can populate their data in clouds instead of directly serving users. The cloud also allows providers to delegate important tasks such as manuscript searches. To protect data security, the documents and their indexes are usually encrypted before outsourcing to the cloud for searches. When users need to query assured documents, they first send keywords to the original data provider. The provider then generates encrypted keywords and returns the trapdoors to the user. The user then sends these trapdoors to the cloud.

Upon receiving the trapdoors, the Cloud uses a special search algorithm to decide on a set of desired documents based on the encrypted indexes and given trapdoors. Lastly, the user receives these encrypted search results and uses the private key from the provider to decrypt documents.

This leads to a extensive search time and more network traffic costs when using traditional search schemes. This revise addresses these issues by proposing an efficient Encrypted Data Search (EnDAS) scheme as a mobile cloud service. This innovative proposal uses a lightweight trapdoor compression method, which optimizes the data communication process by reducing the trapdoor's size for traffic efficiency.

In this study, we moreover propose two optimization methods for document search, called the Trapdoor Mapping Table module and Ranked Serial Binary Search algorithm, to rate the search time. Results demonstrate to make easy EnDAS reduces search time by 38% to 50% as well as network traffic by 12% to 45%.

2. RELATED WORK

Newly, many studies have focused on encrypted search schemes to protect data security and improve search efficiency. For data security, we mainly introduce encryption algorithms and noise methods, while for performance efficiency; we mainly introduce search algorithms, including the Boolean keyword search algorithm and the Ranked keyword search algorithm.To address inefficient two network round trips, we utilize a trapdoor mapping table to obtain singe network round trip.

We ascertain a set of severe privacy requirements for such a secure of various multi-keyword semantics, we choose the efficient similarity measure of "coordinate matching", as many matches as possible, to capture the significance of data documents to the search inquiry. We additional use "inner product similarity" to quantitatively assess such similarity measure. we first propose a basic idea for the MRSE based on secure inner product computation, and then give two significantly improved MRSE schemes to achieve various strict privacy requirements in two different risk models. To achieve search efficiency, we choose a ranked keyword search algorithm and optimize it with binary tree principle. In addition, to address inefficient two network round trips.

3. ENDAS PROPOSE

The propose of the EnDAS system and retreated trapdoor making process in EnDAS. Compared the EnDAS system figure 1 with traditional system.

The main difference is with the aim of set of connections traffic is compact by a single round trip information exchange and the trapdoor compression method; and the search time is reduced by the RSBS algorithm and the TMT module; and the computing burden for generating trapdoors is also off loaded by the TMT module.

Above mentioned performance benefits are enabled by a retreated trapdoor generation process and a retreated search algorithm.

3.1 Structural Design of the EnDAS System

Figure 3 shows the search flow in EnDAS system.The trapdoor generation process and the cloud search algorithm are retreated to reduce search delay and network traffic. For trapdoor generation, EnDAS stores a precompiled Trapdoor Mapping Table in mobile devices, which maps common English words to corresponding trapdoors.

When the mobile device initiates a search request, the trapdoor is looked up from the table instead of being requested from the provider. This optimization saves one network round trip for the trapdoor generation.

Furthermore, EnDAS also provides new algorithms to optimize and compress trapdoors to reduce network traffic to transmit trapdoors will elaborate the details of the EnDAS trapdoor generation process. For the search algorithm, EnDAS proposes to leverage a binary tree structure to reduce the lookup costs and thus improve the search responsiveness.

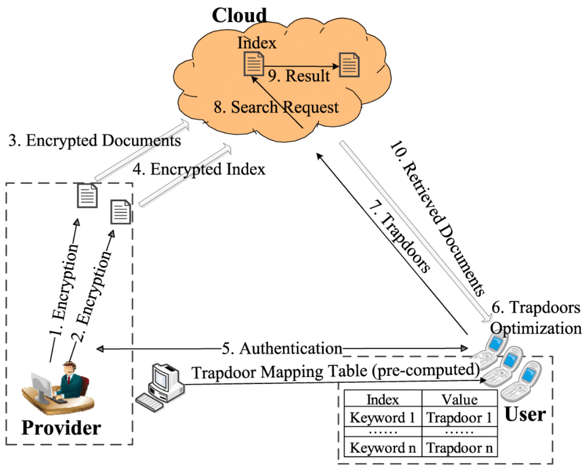


Figure 1: EnDAS system over mobile cloud

New algorithms to optimize and compress trapdoors to reduce network traffic to transmit trapdoors will elaborate the details of the EnDAS trapdoor generation process. For the search algorithm, EnDAS proposes to leverage a binary tree structure to reduce the lookup costs and thus improve the search responsiveness would further explain the details.

4. Retrofitted Trapdoor Generation Process

The retreated trapdoor generation process is described in this process includes the trapdoor mapping table and the trapdoor compression algorithm.

4.1 Summary

With retreated trapdoor generation process it is not necessary for an authenticated user calculate pure trapdoors .After a keyword is stemmed, an user can just query the trapdoor mapping table for the trapdoors, Since the trapdoor mapping table stores the information needed for mapping and search, the heavy computation for generating trapdoors are not needed to be conducted online. This not only avoids the recalculation if the term is found, but also reduces the number of necessary round trips from two to one. It is unavoidable that the trapdoors of some keywords have not been stored in the trapdoor mapping table in advance. In this case, the keyword is encrypted by the user. Then the newly retrieved or generated pure trapdoor is added with some noises from a noise set Θ, to prevent the cloud from examining the same trapdoors.

A lightweight trapdoor compression method is used to extract each trapdoors characteristic bit, record as well as accumulate location of each characteristic bit in order, and transmit the compressed trapdoor to the cloud. Since these characteristic bits only occupy a small proportion in this trapdoor, the compressed trapdoor will lead to additional reduced traffic cost for transmitting the trapdoors to the cloud. .

4.2 PERFORMANCE TRAPDOOR MAPPING TABLE MODULE

We found that there was a long calculation time from building the trapdoor on the provider side. In a traditional system, the calculation of generating a trapdoor of a given keyword is constituted by term stemming, encryption and adding noise by the provider. Among these three steps, it is that the time of encryption stands for a significant proportion of the total trapdoor calculation time displays three columns, denoting the total calculation time for generating trapdoors large amount of frequently-used trapdoors calculated offline. The key for this trapdoor mapping table is a term from stemmed keywords, while its value corresponds to encrypted terms. Next we analyzed the availability of the TMT module. According to our measurement, we found that in 30,000 trapdoors, the size of more than 82% of trapdoors ranges from 30 to 90 bytes. That is, encrypted keywords have a small size. So we selected 5,900 different words as keywords to be encrypted, and then stored them in TMT module. We achieve that the actual size of TMT was about, according to our measurement. Although some rare words are not in the TMT module. Users rarely search documents with them, and therefore we can fully ignore these words indicates that this TMT requires only one transfer to the user, while its size is smaller than the size of one noised trapdoor from the provider to the user. That is, the TMT module saves not only traffic but also search time. TMT module does not require the provider to compute trapdoors through expensive communication between the provider and the user, while it only requires the user to look up trapdoors, avoiding re-computing trapdoors. It reduces network round trips for trapdoor generation from two to one. We analyze the performance of EnDAS in search time when generating trapdoors. Utilizing TMT module, EnDAS has only one network round trip used to search target documents which displays three columns, denoting a single keyword, two keywords and three keywords respectively.

5. EFFICIENT SEARCH ALGORITHM

5.1 Document Index Construction

The cloud uses the indexes provided by the provider to quickly search documents. The provider is responsible for constructing document indexes and sends to the cloud. In general, two important matrices are commonly used to generated the index of documents. The Term-Frequency matrix denotes the frequency of each term in documents. The Inverted Document Frequency matrix depicts the significance of rare terms that are used to distinguish documents. The multiplication of these two matrices, which produces the score matrix A.The matrix A will be encrypted and out sourced to the cloud, rather than traditional TF matrix and IDF matrix. This avoids multiplication operation (TF×IDF) when searching documents score in the cloud. Suppose we have N documents and T terms, matrix A is a N-by T matrix. Each element RSt, c stands for the relevance score of term t in document c, for a particular document c, c ∈ {1,...N} and a term t, t ∈ {1,...,T}. We use the column vectors Ic of matrix A as the index for a particular document.

5.2 Index Slicing

After the plain-text document indexes are produced, the provider then divides each index into s slices s (s ≤ T) according to the score value. We elaborate this process as follows. According to score value, we divide the index Ic into s slices, and each slice has a normalized score value. And terms in one slice, such as the slice Slicec, are given a same score value as the normalized score of this slice.

According to this principle, we define the slice Slicec as the jth slice of the index Ic, so its score scope Scopej,c = [max(RSt,c)−min(RSt,c) s ×(j −1), max(RSt,c)−min(RSt,c) s ×j](t ∈ {1,...,T},c ∈ {1,...N},1 ≤ j ≤ s), where max(RSt,c) denotes the maximum score of the matrix A, min(RSt,c) denotes the minimum score of the matrix A.

5.3 Index Encryption

Utilizing this FAH algorithm, we encrypt slices of each index. The detailed encryption process for one slice Slicec of the index Ic is that encrypting l-bit term t in Slicec is used by the hash function H(), and mapping l-bit encrypted term τt into r-bit optimized term ¯ τt is by the mapping function G(), where l = d × r; and then accumulating all the r-bit optimized terms together. Finally we get the encrypted slice Slice0 c. In this way, we can encrypt the index Ic by accumulating all the slices and obtain the encrypted index I0 c equals accumulating all the optimized terms in this document.

5.4 Binary Search Tree for Indexes

We propose to generate a binary search tree for indexes in order to accelerate the search time. By utilizing the FAH algorithm, each document's index is processed as a hash code comprised by accumulated terms. With time, we can testify if a given trapdoor appears in this document. If so, we will further identify the slice within the document, which contains the given trapdoor. To accelerate the entire procedure**,** we construct a binary tree. In this data structure, the top level is a hash code comprised by all accumulated terms. On the second level, each descendant only contains the accumulated terms of half of the index. Further down, all descendants contain the accumulated terms of half of that from its parent. With such structure, the height of the tree is at most s (the number of slices in the index) and thus the search efficiency is O(s).

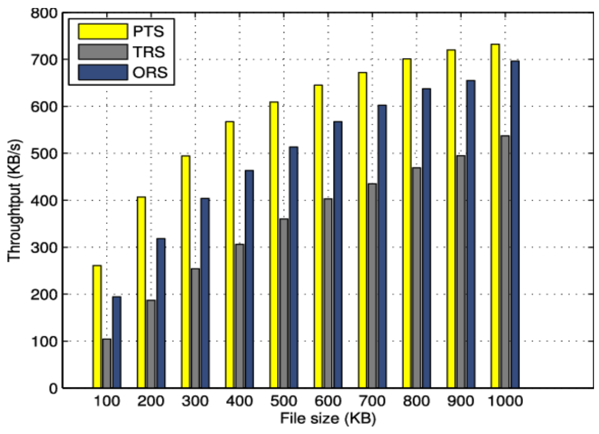
5.5 RSBS Algorithm

Upon receiving a trapdoor , the cloud would perform a privacy preserving search from the indexes provided by the provider. Then it selects top-k documents that contain the given search keywords. This process is achieved by using the RSBS algorithm. The RSBS algorithm aims to find the top-k documents that best match the search keywords provided by the user. To this end, it maintains a score array for each document. The main idea is to compute accumulated scores for each document.

5.5.1Time Complexity Analysis

The RSBS algorithm traverses through all documents and all keywords in user's search request, which makes the inner-most body iterated for eN times. Here e represents the number of keywords provided by the user, and N represents the number of documents. In each iteration, the binary search will be executed and its time complexity is O(log(s)) . Thus RSBS algorithm has a time complexity of O(eNlog(s)). Comparing with traditional systems with a time complexity of O(eNs), RSBS can effectively reduce the search time by utilizing the binary search. In practice, RSBS algorithm can be further parallelized to compute eN binary searches concurrently, which could further reduce its actual execution time. To this end, it maintains a score array for each document.

5.5.2 GRAPHICAL OUTPUT IN TIME COMPLEXITY



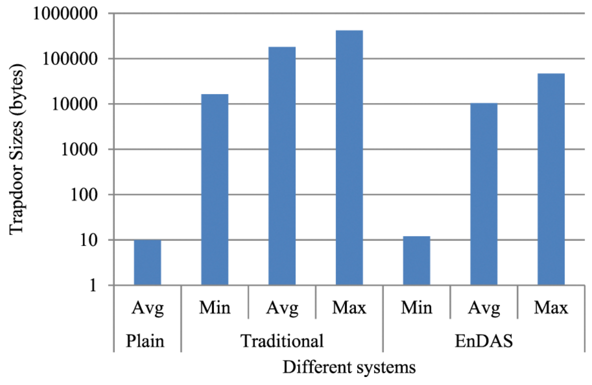


Figure 2: Performance Analysis

6. CONCLUSION

In this work, we proposed a novel encrypted search system EnDAS over the mobile cloud, which improves network traffic and search time efficiency compared with the traditional system. We started with a thorough analysis of the traditional encrypted search system and analyzed its bottlenecks in the mobile cloud: network traffic and search time inefficiency. Then we developed an efficient architecture of EnDAS which is suitable for the mobile cloud to address these issues, where we utilized the TMT module and the RSBS algorithm to cope with the inefficient search time issue, while a trapdoor compression method was employed to reduce network traffic costs.

7. REFERENCES

[1] D. Huang, “Mobile cloud computing,” IEEE COMSOC Multimedia Commune. Tech. Committee (MMTC) E-Letter, vol. 6, no. 10, pp. 27– 31, 2011.

[2] N. Cao, C. Wang, M. Li, K. Reno, and W. Lou, “Privacy-preserving multi-keyword ranked search over encrypted cloud data,” in Proc. Int. Conf. Computes Common. (INFOCOM), Apr. 2011, pp. 829–837.

[3] C. Wang, N. Cao, J. Li, K. Reno, and W. Lou, “Secure ranked keyword search over encrypted cloud data,” in Proc. IEEE Int. Jun. 2010, pp. 253–262.

[4] C. Gentry and S. Halve, “Implementing gentrys fullyhomomorphic encryption scheme,” in Advances in Cryptology– EUROCRYPT 2011, 2011, pp. 129–148.

[5] C. ¨Orense and E. Sava’s¸, “Efficient and secure ranked multikeyword search on encrypted cloud data,” in Proc. Joint EDBT/ICDT Workshops, Mar. 2012, pp. 186–195.

[6] Gartner, “Worldwide traditional pc, tablet, ultra mobile and mobile phone shipments on pace to grow 7.6 percent in 2014,” http://www.gartner.com/newsroom/id/2645115.

[7] K. D. Bowers, A. Juels, and A. Oprea, “Hail: a high-availability and integrity layer for cloud storage,” in Proc. ACM Conf. Compute. Commune. Secure. (CCS), Dec. 2009, pp. 187–198.

[8] J. Li, R. Ma, and H. Guan, “Tees: An Efficient search scheme over encrypted data on mobile cloud,” IEEE Trans. Cloud Compute., Feb. 2015. [36] N. Cao, C. Wang, M. Li, K. Reno, and W. Lou, “Privacy-preserving multi-keyword ranked search over encrypted cloud data,” IEEE Trans.ParallelDistrib.Systems,vol.25,no.1,pp.222–233,Jan.2014.

[9] C. Wang, N. Cao, K. Ren, and W. Lou, “Enabling secure and efficient ranked keyword search over outsourced cloud data,” IEEE Trans. Parallel Distrib. Systems, vol. 23, no. 8, pp. 1467–1479, 2012.

[10] B. Wang, S. Yu, W. Lou, and Y. T. Hoe, “Privacy-preserving multikeyword fuzzy search over encrypted data in the cloud,” in Proc. Int. Conf. Computes. Communes. (INFOCOM), Apr. 2014, pp. 2112– 2120.