SECURE DATA TRANSFER AND DELETION FROM COUNTING BLOOM FILTER IN CLOUD COMPUTING

[#1]M.DIVYA, *M.Tech Student, Department of CSE,*
[#2]*Mr.N.VENKATESHWARAN, Associate Professor, Department of CSE,*
[#3]Dr.M.SUJATHA, *Associate Professor, Department of CSE,*
[#4]Dr.R.JEGADEESAN, *Professor & HOD, Department of CSE*
JYTOHISHMATHI INSTITUTE OF TECHNOLOGY & SCIENCE, KARIMANAGAR,TS.

ABSTRACT: With the rapid development of cloud storage, an increasing number of data owners prefer to outsource their data to the cloud server, which can greatly reduce the local storage overhead. Because different cloud service providers offer distinct quality of data storage service, e.g., security, reliability, access speed and prices, cloud data transfer has become a fundamental requirement of the data owner to change the cloud service providers. Hence, how to securely migrate the data from one cloud to another and permanently delete the transferred data from the original cloud becomes a primary concern of data owners. To solve this problem, we construct a new counting Bloom filter-based scheme in this paper. The proposed scheme not only can achieve secure data transfer but also can realize permanent data deletion. Additionally, the proposed scheme can satisfy the public verifiability without requiring any trusted third party. Finally, we also develop a simulation implementation that demonstrates the practicality and efficiency of our proposal.

Key words — Cloud storage, Data deletion, Data transfer, Counting Bloom filter, Public verifiability.

1. INTRODUCTION

As a new computing paradigm, cloud computing is the fusion and development of parallel computing, distributed computing and grid computing. Cloud storage is one of the most attractive services offered by cloud computing, which can provide users with convenient data storage services and business access services by integrating a large number of distributed storage devices in network together. In cloud storage, users can outsource their data to the cloud server, which can greatly reduce the local hardware/software overhead and human resources investments. Due to the attractive advantages, cloud storage has been widely applied in the daily life and work. As a result, more and more resource constraint users, including individuals and corporations prefer to embrace cloud storage service. Despite tremendous advantages, cloud storage inevitably suffers from some new security problems because of the separation of outsourced data ownership and management, such as data confidentiality, data integrity, data availability and data deletion. These problems, specifically for data deletion, if not solved well, may impede the acceptance of cloud storage to the public. As the last phase of the data life cycle, data deletion directly determines whether the data life cycle can come to an end favourably, which is very important for data security and privacy-preserving. However, data deletion attracts much less attention compared with data integrity, which has been well studied and solidly solved. Although some verifiable deletion schemes have been proposed for outsourced data in cloud computing environment, there are still some problems and challenges that urgently need to be solved solidly.

To realize secure data migration, an outsourced data transfer app, Cloudsfer, has been designed utilizing cryptographic algorithm to prevent the data from privacy disclosure in the transfer phase. But there are still some security problems in processing the cloud data migration and deletion. Firstly, for saving network bandwidth, the cloud server might merely migrate part of the data, or

even deliver some unrelated data to cheat the data owner. Secondly, because of the network instability, some data blocks may lose during the transfer process. Meanwhile, the adversary may destroy the transferred data blocks. Hence, the transferred data may be polluted during the migration process. Last but not least, the original cloud server might maliciously reserve the transferred data for digging the implicit benefits. The data reservation is unexpected from the data owners' point of view. In short, the cloud storage service is economically attractive, but it inevitably suffers from some serious security challenges, specifically for the secure data transfer, integrity verification, verifiable deletion. These challenges, if not solved suitably, might prevent the public from accepting and employing cloud storage service.

2. LITERATURE REVIEW

A. Filtering Bloom

Bloom Data bases can be used as a room-efficient alternative for membership inquiry problems and other network applications for the fast match of IP routing. Standard bloom filters provide a compact overview. It is possible to search and insert. You can change the False Positive Filter (FP) rate. The result of the query is "no" or "presumably" "yes." The compromise lies between efficiency and space for floral filters. K-hash functions are provided. Every bit is "0" at the start. Add the item bit by bit in position K-hazh and set k-bits to 1. In the search, the k-bits for the hash functions are checked. The request could be correct if all k-bits are '1.' The item may be available. However, this may be wrong, but Bloom Filters have no wrong negative elements.

B. Variants of Bloom Filter

The overhead query is slightly larger due to the Bloom-1 or Fast Bloom filter memory dimensions. B. The Bloom Scalable Filter provides an effective Dynamic Bloom filter rate of 21.3% and increases CPU time logarithmically, not linearly. Some bloom filters can successfully validate large database sets to validate each sub-set of the DBA-controlled database. These flowers are high-performing because they can be accessed at the same time. A floromide variant is more resistant to collision than nonlinear avalanche bits for quick and scalable applications such as safe wireless broadcast. Compact mapping with LSFR arrays is performed and the entire design can be reconfigured by implementing FGPA.

With successive prime numbers and a unique hacha operation, the base hatch and modular operation are used to minimise the overview risk. A small set variant for Bloom Filters has higher space efficiency and a part support for false positive removals of less than 2.8 per cent. Another approach includes a Bloom Filter based tree, whereby a portion of interior nodes can be searched by structured tree data, by assigning the flower filter sections to the data set. I have a Bloom Tree Filter structured, a tree depth, time is constantly complicated. The Cuckoo filters store Bit fingerprints in hash tables that allow the dynamic addition or deletion of items. Small spaces are also available (when the FP rate is below 3%) and the search performance increases. A florometric variable called the Par-BF(Divided BF) contains a low overall performance balance and fast formulas for the identification of key dynamic data parameters. It improves rapid simultaneous interaction with a waste collector policy generally lower than DBF and surpasses DBF (dynamic flower filter). The performance metrics of Bloom Filter include memory requirements, false positive features and request overhead. For applications not tolerable to misplaceable but false negative applications, flower filters are valid.

3. ANALYSIS FOR BLOOM FILTER TECHNICAL

Bloom Filter (BF) is a binary string vector data structure proposed by Howard Bloom in 1970, it has a very good space and time efficiency, its function is to be used to detect an element weather is a member of the set. As a simple information representation scheme, Bloom Filter can well meet the demand for resource efficient interaction and research on the high-speed network, particularly suitable for distributed systems in data storage and efficient query.

A. Algorithm principle

The standard Bloom Filter is described as follows: data set S={s1,s2, …,si ,…,sn}(1≤i≤n), there are n elements, through the hash functions H1,H2, …,Hj ,…,Hk(1≤j≤k) which range from 0 to m-1 mapped to the bit sequence of length m BF vector, hash functions are independent. Principle as shown in Fig.1.

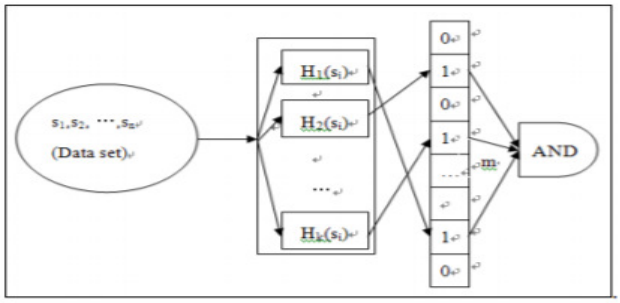

Fig.1 Standard Bloom Filter

Standard Bloom Filter can implement the insertion and query operation of the elements, there are false positive questions during the query, and the schematic diagram of the element insertion and query operation is shown in Fig.2.

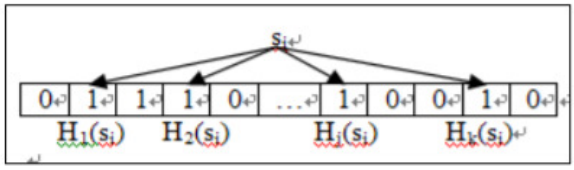

Fig.2 Insertion and Query of element Bloom Filter

elements separately, and then set the corresponding BF bit string values to 1.When querying operation, also firstly called k hash functions hash the elements separately, and then query the BF bit string whether the value of the operation result corresponding bit is 1, as long as there value 0, the element is not set in.

when $k = \left\lfloor (\ln 2)(m/n) \right\rfloor$

The error rate of the query is minimum. For example, when m=1280 bits, n=100, K takes 9 can make the false positive rate get to the minimum 0.0021. The impact of such a low rate of miscarriage of justice in the network monitoring system will be very small.

B. Units Improvement research and typical application

Since such an excellent data structure raised, improvement research based on it has been carried, a lot of research have been emerged. There are more typical of the extended algorithm: Counting bloom filter solved the question which elements cannot be deleted; Compressed Bloom filter using arithmetic coding technique and the ultimate entropy to compress BF vector as a message transfer in distributed system, it can obtain higher compression rate and the error rate will be reduced; others improvement research are: Spectral Bloom filter query algorithm, Split type Bloom filter query algorithm, and so on. A lot of improvement research shows that: Bloom Filter is a very potential technology, and its application is very wide.. Bloom Filter is mainly played two roles in the following practical application:

(1) Due to the Bloom Filter has the characteristics of saving space, which can be as a compression of the small size of the data collection used to replace the original large-scale data, complete the judgement weather the element is in the data set, to express large data sets and improve the searching efficiency. This kind of application mode mainly concentrates in the aspects such as database operation, Dictionary inquiry and file operation, resource route, packet route, network intrusion detection and so on.

(2) The large scale raw data set is abstracted as abstract information with Bloom Filter algorithm, and then the summary information is transmitted in other distributed nodes such as database. The most typical of this application is the transfer of the contents of the file directory information to each host in the distributed system.

Will file directory information in form of a bloom filter, storage to the DRAM in node, because of the concise Bloom Filter storage, which can not only speed up file search process, and can reduce the interaction of bandwidth consumption.

4. SIMULATION EXPERIMENT

Time cost of data outsourcing. In cloud storage, the data is outsourced to the cloud server. To protect the outsourced data confidentiality, the user encrypts the data before uploading. In this experiment, we fix the number of outsourced data blocks n = 1, 000 for simplicity. Meanwhile, we increase the size of encrypted plaintext from 1 MB to 10 MB with a step for 1 MB. Then, test the approximate time cost, as demonstrated in Figure. We can easily find from Figure 3 that the time

cost linearly increases with the size of encrypted plaintext. Meanwhile, the time overhead of our proposed scheme is a little larger than that of scheme, but it is less than that of scheme and it is absolutely acceptable. For example, when the size of outsourced file reaches 10 MB, our proposed scheme costs about 52 milliseconds, while scheme costs 41 milliseconds and scheme costs 85 milliseconds. Moreover, the growth rate of scheme is highest. Therefore, we can think that our proposed scheme is still very efficient in data outsourcing process.

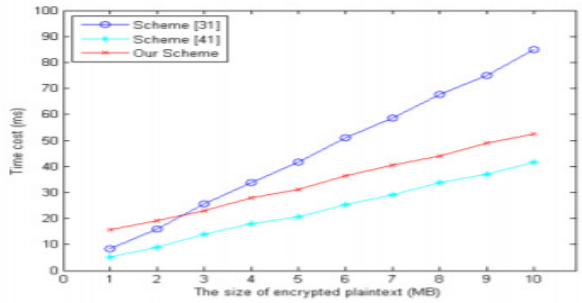

Fig: Time cost in data outsourcing

Time cost of storage checking. Due to the lack of trust in cloud server, the user wants to verify the data storage result after outsourcing. In our proposed scheme, the user needs to execute one signature verification operation and (k + 3)n hash calculations. While the existing scheme needs to perform n modular exponentiation computations and 2n bilinear pairing calculations. Moreover, scheme cannot achieve provable storage. In our experiment, we increase the number of outsourced data blocks (i.e., number n) from 100 to 1,000 with a step for 100. Then, we test the approximate time overhead, as demonstrated in Figure. We can easily find that the time cost increases with number n. However, the growth rate of our proposed scheme is relatively lower and it is so small that it can be negligible. Meanwhile, the existing scheme costs much more time overhead than our proposed scheme. That is, our proposed scheme is much efficient in data storage checking process.

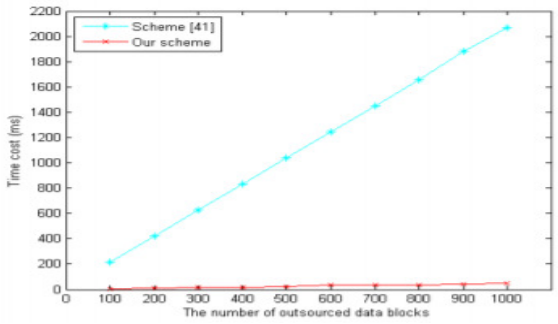

FIGURE : Time cost in storage checking

Time cost of data deletion. When some outsourced data blocks will not be needed anymore, the user is willing to permanently delete them for protecting data privacy and saving storage expensive. In our proposed scheme, the computation overhead in this step contains two signature generation operations, one signature verification operation and (k + 3)l hash calculations. However, scheme needs to execute l signature generation operations. Then, scheme needs to execute n modular exponentiation computations, l + 2 signature generation computations and one signature verification operation. In this experiment, we fix the number n = 1, 000 and a file is viewed as a data block in scheme for simplicity. Then, we increase the number of deleted data blocks from 10 to 100 and Figure demonstrates the efficiency comparison. We can easily see from Figure that the time overhead increases with the number of deleted data blocks. Meanwhile, the growth rate of our proposed scheme is relatively lower than these of scheme and scheme. Meanwhile, the time cost of our proposed scheme is least. Therefore, we can think that our proposed scheme is more efficient than schemes in data deletion process.

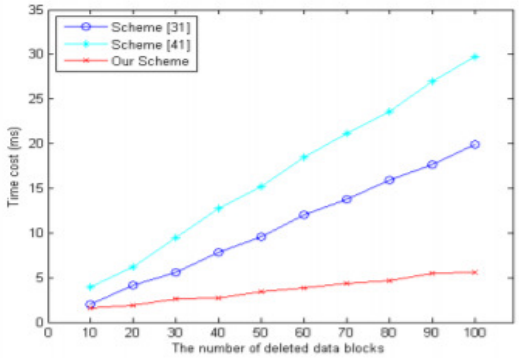

FIGURE: Time cost in data deletion

Time cost of deletion checking. After the cloud server deleting the unnecessary data blocks, the

user checks the data deletion result by verifying the corresponding deletion evidence. In this process, the computation overhead of our proposed scheme contains one signature verification operation and kl hash calculations, while scheme needs l modular exponentiation computations and l+2 signature verification operations. Moreover, scheme needs l signature verifications operations. Similarly, we increase the number of deleted data blocks from 10 to 100 with a step for 10 and test the approximate time overhead, as demonstrated in Figure. We can easily find from Figure 6 that the growth rates of scheme and scheme are relatively higher than that of our proposed scheme. Moreover, our proposed scheme costs least time overhead. For example, when the number of deleted data blocks reaches 100, our proposed scheme costs about 4.4 milliseconds, while scheme costs 36 milliseconds and scheme needs about 55 milliseconds. As a result, we can believe that our proposed scheme is very efficient to check the data deletion result.

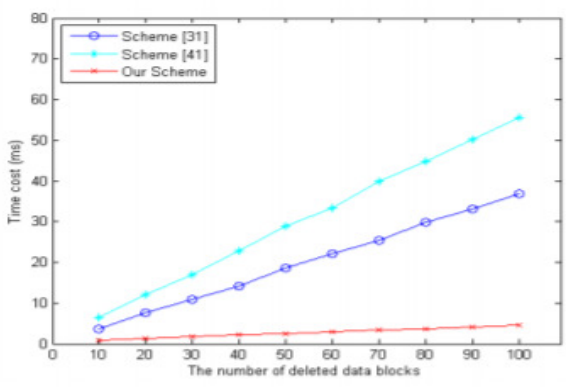

FIGURE Time cost in deletion checking

5.CONCLUSION

In cloud storage, the data owner does not believe that the cloud server might execute the data transfer and deletion operations honestly. To solve this problem, we propose a CBF-based secure data transfer scheme, which can also realize verifiable data deletion. In our scheme, the cloud B can check the transferred data integrity, which can guarantee the data is entirely migrated. Moreover, the cloud A should adopt CBF to generate a deletion evidence after deletion, which will be used to verify the deletion result by the data owner. Hence, the cloud A cannot behave maliciously and cheat the data owner successfully. Finally, the security analysis and simulation results validate the security and practicability of our proposal, respectively.

Future work Similar to all the existing solutions, our scheme considers the data transfer between two different cloud servers. However, with the development of cloud storage, the data owner might want to simultaneously migrate the outsourced data from one cloud to the other two or more target clouds. However, the multi-target clouds might collude together to cheat the data owner maliciously. Hence, the provable data migration among three or more clouds requires our further exploration.

REFERENCES:

[1] C. Yang and J. Ye, “Secure and efficient fine-grained data access control scheme in cloud computing”, *Journal of High Speed Networks*, Vol.21, No.4, pp.259–271, 2015.

[2] X. Chen, J. Li, J. Ma, *et al*., “New algorithms for secure outsourcing of modular exponentiations”, *IEEE Transactions on Parallel and Distributed Systems*, Vol.25, No.9, pp.2386–2396, 2014.

[3] P. Li, J. Li, Z. Huang, *et al*., “Privacy-preserving outsourced classification in cloud computing”, *Cluster Computing*, Vol.21, No.1, pp.277–286, 2018.

[4] B. Varghese and R. Buyya, “Next generation cloud computing: New trends and research directions”, *Future Generation Computer Systems*, Vol.79, pp.849–861, 2018.

[5] W. Shen, J. Qin, J. Yu, *et al*., “Enabling identity-based integrity auditing and data sharing with sensitive information hiding for secure cloud storage”, *IEEE Transactions on Information Forensics and Security*, Vol.14, No.2, pp.331–346, 2019.

[6] R. Kaur, I. Chana and J. Bhattacharya J, “Data deduplication techniques for efficient cloud storage management: A systematic review”, *The Journal of Supercomputing*, Vol.74, No.5, pp.2035–2085, 2018.

[7] Cisco, “Cisco global cloud index: Forecast and methodology, 2014–2019”, available at:

https://www.cisco.com/c/en/us-/solutions/collateral/service-provider/global-cloud-index-gci/ white-paper-c11-738085.pdf, 2019-5-5.

[8] Cloudsfer, "Migrate & backup your files from any cloud to any cloud", available at: *https://www.cloudsfer.com/*, 2019-5-5.

[9] Y. Liu, S. Xiao, H. Wang, *et al*., "New provable data transfer from provable data possession and deletion for secure cloud storage", *International Journal of Distributed Sensor Networks*, Vol.15, No.4, pp.1–12, 2019.

[10] Y. Wang, X. Tao, J. Ni, *et al*., "Data integrity checking with reliable data transfer for secure cloud storage", *International Journal of Web and Grid Services*, Vol.14, No.1, pp.106–121, 2018.

[11] Y. Luo, M. Xu, S. Fu, *et al*., "Enabling assured deletion in the cloud storage by overwriting", *Proc. of the 4th ACM International Workshop on Security in Cloud Computing*, Xi'an, China, pp.17–23, 2016.

[12] C. Yang and X. Tao, "New publicly verifiable cloud data deletion scheme with efficient tracking", *Proc. of the 2th International Conference on Security with Intelligent Computing and Big-data Services*, Guilin, China, pp.359–372, 2018.