Balancing the Load for Preserving Privacy to Big Data in Cloud

Akhila[1], Pradeep[2], Priyanka[3], Dr. Ushasakthivel[4]
[1, 2, 3]UG students, department of computer science, Rajarajeswari college of engineering
India
[4] Professor and HOD, department of computer science, Rajarajeswari college of engineering
India

***Abstract*- To simplify data management and to reduce maintenance cost many users and companies move their data to cloud storage in the era of big data. Security and privacy are the major problems since third party cloud service providers are not trusty services. Here, data contents are protected by encryption, the access patterns that contain useful information can be exposed to clouds or malicious attackers. The ORAM algorithm is applied to enable privacy-preserving access to big data that are deployed in distributed file systems. Since, the ORAM algorithm would lead to serious access load unbalance among storage servers, a data placement problem is used to achieve load balanced storage system with improved availability and responsiveness. A low complexity algorithm is used to deal with large scale problem size with respect to big data. Extensive simulations are conducted to show that results are close to the optimal solution.**

I. INTRODUCTION

Big data is an evolving term that describes any voluminous amount of structured, semi-structured and unstructured data that has potential to be mined for information. It has emerged in various fields including science, engineering and commerce. For example, the amount of photos currently stored by Face book is over 20 petabytes, and it continues to grow with 60 terabytes each week. In the era of big data, cloud becomes a perfect candidate for data storage by providing virtually unlimited storage that can be accessed over network. The pay-as-you-use model is used by the user to simplify their data management and reduce data maintenance cost in Google Drive, Drop box and Amazon S3 by outsourcing large volumes of data to cloud storage. However, some users and companies still hesitate to move their data to cloud because of security and privacy concerns. Although data confidentiality is protected by encryption, it is insufficient because access patterns can also leak important information.

For instance, over 80% of encrypted email queries can be identified according to access pattern in. The access privacy problem is first addressed by private information retrieval (PIR) technique that allows a user to retrieve a block from a database of N items held by a server that learns nothing about this block. The existing PIR schemes will never be more efficient than a trivial PIR scheme of downloading the entire database. The extremely poor performance of PIR makes it inapplicable in cloud storage with big data. Oblivious RAM (ORAM) is later proposed to hide data access privacy with improved performance. Its basic idea is to periodically reshuffle data blocks stored in an entrusted server such that user access cannot be tracked. Goodrich et al. have proposed an ORAM algorithm with $O(\sqrt{N})$ client storage to achieve $O(\log N)$ amortized cost, i.e., each oblivious read or write leads to $O(\log N)$ data access operations on average. Shi et al. further reduce the client storage to
$O(1)$.

Here, the ORAM algorithm is used to enable privacy-preserving access to big data in clouds. To deal with the challenge of accommodating huge volume of data that continuously grows in high velocity, big data are stored in distributed file systems built upon hundreds or thousands of servers in a single or multiple geo-distributed cloud sites. When ORAM is directly applied on such distributed storage systems, we observe that even if all data blocks are evenly accessed by users, access load on servers would be seriously unbalanced, i.e., lots of data access requests are sent to several servers, but only a few to others. The servers with high load are apt to be system bottleneck or failure points in the system. Motivated by this observation, we exploit the data access characteristics of ORAM, and propose a data placement algorithm to achieve load balance, thus improving overall system availability and responsiveness.

The main contributions of this paper are summarized as follows. First, we study the privacy-preserving data access to big data in an entrusted cloud by applying the ORAM algorithm. In conjunction with encryption, ORAM-based solutions can hide not only data contents but also access patterns from third party cloud service provider and malicious attackers. Second, we investigate a load balance problem in applying ORAM on distributed file systems. This problem is proved to be NP-hard, and formulated as a mixed integer linear programming (MILP) problem. We propose a low-complexity algorithm that can deal with large-scale problem instances with respect to big data. Third, extensive simulations are conducted to show that the performance of our proposed algorithm is close to the optimal solution, and significantly outperforms a random data placement algorithm.

II. RELATED WORK

A. Private Information Retrieval

In the online retrieving valuable information in special databases, it is important to protect the privacy of users preventing the server from knowing the information what the users retrieve. Although a private information retrieval (PIR) scheme allows a user to retrieve a date item from online database while hiding the index of the data item from the database server. However, retrieving valuable information is another problem needs to be solved. Most transactions are completed by credit card through delivering id number and the personal data of the user in the Internet to the server. Then the server will know the privacy of users. It will become a challenge to PIR schemes considering the e-payment need. In this paper, a novel PIR scheme is proposed which uses a secure coprocessor (SC), with e-payment scheme for protecting the privacy of customers in querying valuable information. The scheme will not reveal personal information or buying information of customers in the internet environment, it can really protect the customers' privacy.

B. Cloud Storage

The computing power in a cloud computing environments is supplied by a collection of data centers, or cloud data storages (CDSs) housed in many different locations and interconnected by high speed networks. CDS, like any other emerging technology, is experiencing growing pains. Data integrity checking of data and data structures has grown in importance recently in cloud computing due to the expansion of online cloud services, which have become reliable and scalable. In this paper we propose an integrity layered architecture of a typical cloud based on MAS architecture consists of two main layers cloud resources layer (cloud server-side) and MAS architecture layer (cloud client-side). At the cloud resources layer there exist massive physical cloud resources (storage servers and cloud application servers) that power the CDS. MAS's architecture layers consist of two agents: Cloud Service Provider Agent (CSPA) and Cloud Data Integrity Backup Agent (CDIBA). This layered architecture named as “Cloud Zone”. A prototype of our proposed “Cloud Zone” will be designed using Prometheus Methodology and implemented using the Java Agent Development Framework Security (JADE-S).

C. Wide Area Co-operative Storage with CFS

This paper proposes a multi agent system that carries out cooperative works. To realize the cooperative works, we use fuzzy associative memory organizing unit systems (FAMOUS) and conceptual fuzzy sets (CFS). By using this proposed method, each agent robots can decide its own behavior for the situation of its environment. We apply this system to intelligent transportation system (ITS) and simulate

D. Security Monitoring System

In this paper, we describe the problem of checking the integrity of log in monitoring system for forensics investigation. Existing frameworks and solutions do not provide a satisfactory result to solve the problem. They either require a mass amount of storage overhead to store the hash values of the events or may not be able to match the situation in an effective way if some events have been modified. We propose an efficient hashing scheme with Shifted Transversal Design Group Testing algorithm to calculate hash values for all events in a log file as the integrity proof and precisely locate the events which have been corrupted. Experimental results show that the storage overhead can be significantly decreased by adopting the scheme.

E. Oblivious RAM

In this study, the authors design efficient protocols for a number of `oblivious decision program (DP) evaluation' problems. Consider a general form of the problem where a client who holds a private input interacts with a server who holds a private DP (e.g. a decision tree or a branching program) with the goal of evaluating his input on the DP without learning any additional information. Many known private database query problems such as symmetric private information retrieval and private keyword search can be formulated as special cases of this problem. Most of the existing works on the same problem focus on optimizing communication. However, in some environments (supported by a few experimental studies), it is the computation and not the communication that may be the performance bottleneck. In this study, we design `computationally efficient' protocols for the above general problem, and a few of its special cases. In addition to being one-round and requiring a small amount of work by the client (in the RAM model), the proposed protocols only require a small number of exponentiations (independent of the server's input) by both parties. The proposed constructions are, in essence, efficient and black-box reductions of the above problem to 1-out-of-2 oblivious transfer. It is proved that the proposed protocols secure (private) against `malicious' adversaries in the standard ideal/real-world simulation-based paradigm.

F. Simulation on Oblivious RAM

In this study, the authors design efficient protocols for a number of `oblivious decision program (DP) evaluation' problems. Consider a general form of the problem where a client who holds a private input interacts with a server who holds a private DP (e.g. a decision tree or a branching program) with the goal of evaluating his input on the DP without learning any additional information. Many known private database query problems such as symmetric private information retrieval and private keyword search can be formulated as special cases of this problem. Most of the existing works on the same problem focus on optimizing communication. However, in some environments (supported by a few experimental studies), it is the computation and not the communication that may be the performance bottleneck. In this study, we design `computationally efficient' protocols for the above general problem, and a few of its special cases. In addition to being one-round and requiring a small amount of work by the client (in the RAM model), the proposed protocols only require a small number of exponentiations (independent of the server's input) by both parties. The proposed constructions are, in essence, efficient and black-box reductions of the above problem to 1-out-of-2 oblivious transfer. It is proved that the proposed protocols secure (private) against `malicious' adversaries in the standard ideal/real-world simulation-based paradigm.

III. IMPLEMENTATION

ORAM allows a trusted processor to use an entrusted RAM. Most existing ORAM solutions use the basic memory structure suggested by Ostrovsky's Hierarchical Scheme. The ORAM is arranged in a series of progressively larger caches. Each cache consists of a hash table of buckets. When a block is requested, the algorithm checks a bucket at each level of the hierarchy. If the block is found, the search continues for a dummy block such that the location of his desired block is hidden. Finally, the block is reinserted into the top-level cache. When a cache is close to over flowing, it is obliviously shuffled into the cache below it.

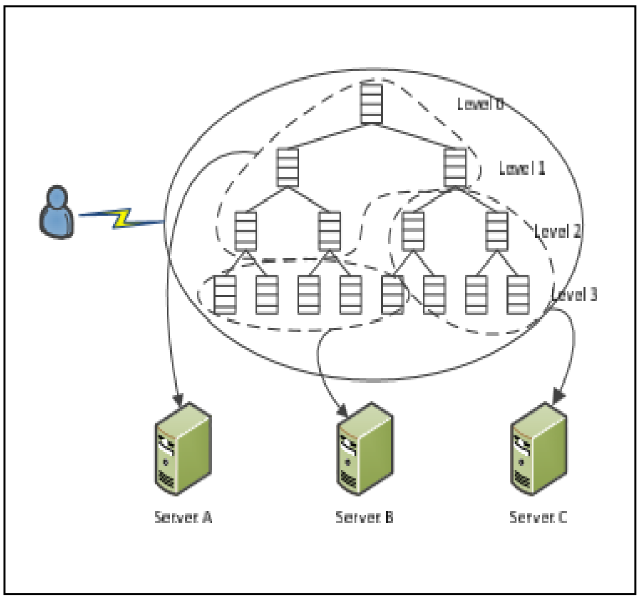

Figure1: ORAM based cloud storage.

Consider a client that would like to store and retrieve its big data in cloud that is honest but curious. In other words, the cloud cannot tamper with or modify the data, but could learn information about the data. The data are divided into blocks, each of which is identified by a unique address. For example, a typical value of block size is 64KB or 256KB. Data stored on cloud are organized as a tree, where each node is referred to as a bucket that stores several data blocks. An example of a binary tree structure is shown in Fig. 1. Note that any arbitrary tree structure is applicable in ORAM. Following the work in , we translate each read or write operation into two primitives ReadAndRemove and Add that are defined as follows. ReadAndRemove(u): given an address u specified by the client, the cloud returns the corresponding data block, and removes it from storage. Add (u, d): the client writes block d to address u at the client storage. With above two primitives, each read(u) operation can be replaced by a ReadAndRemove(u) followed by an Add(u, d) that writes the same data block back to address u. Similarly, to implement a write(u,d) operation, we conduct a redundant ReadAndRemove(u) before Add(u, d). Although the number of access operations is doubled in ORAM, it prevents the entrusted cloud from distinguishing read and write operations.

IV.CONCLUSION

The proposed work applies the ORAM algorithm to achieve privacy-preserving access to big data in clouds. On applying ORAM observe a load unbalance phenomenon after deploying ORAM-based storage to multiple servers, which motivates to investigate a data placement problem to achieve load balance. This problem is proved to be NP-hard. The proposed work propose a low-complexity algorithm to solve this problem with respect to large data volumes. Simulation results show that the performance of our proposed algorithm is close to optimal solution, and it outperforms a random data placement algorithm.

V.REFERENCES

[1] M. Islam, M. Kuzu, and M. Kantarcioglu, “Access pattern disclosure on searchable encryption: Ramification, attack and mitigation,” in *Network and Distributed System Security Symposium*, 2012.

[2] B. Chor, E. Kushilevitz, O. Goldreich, and M. Sudan, “Private information retrieval,” *Journal of the ACM (JACM)*, vol. 45, no. 6, pp. 965–981, 1998

[3] R. Sion and B. Carbunar, “On the computational practicality of private information retrieval,” in *Proceedings of the Network and Distributed Systems Security Symposium*, 2007, pp. 2006–06.

[4] M. T. Goodrich and M. Mitzenmacher, “Mapreduce parallel cuckoo hashing and oblivious ram simulations,” *CoRR*, vol. abs/1007.1259, 2010.

[5] E. Shi, T.-H. H. Chan, E. Stefanov, and M. Li, “Oblivious ram with o ((logn) 3) worst-case cost,” in *Advances in Cryptology–ASIACRYPT 2011*. Springer, 2011, pp. 197–214.

[6] K. D. Bowers, A. Juels, and A. Oprea, “Hail: A high-availability and integrity layer for cloud storage,” in *Proceedings of the 16th ACM Conference on Computer and Communications Security*, 2009, pp. 187–198.

[7] F. Dabek, M. F. Kaashoek, D. Karger, R. Morris, and I. Stoica, “Widearea cooperative storage with cfs,” in *ACM Symposium on Operating Systems Principles*, 2001, pp. 202–215.

[8] A. Bessani, M. Correia, B. Quaresma, F. Andr´e, and P. Sousa, “Depsky: Dependable and secure storage in a cloud-of-clouds,” in *Proceedings of the Sixth Conference on Computer Systems*, 2011, pp. 31–46.

[9] E. Stefanov, M. van Dijk, A. Juels, and A. Oprea, “Iris: A scalable cloud file system with efficient integrity checks,” in *Proceedings of the 28th Annual Computer Security Applications Conference*, 2012, pp. 229–238.

[10] Z. Wu, M. Butkiewicz, D. Perkins, E. Katz-Bassett, and H. V. Madhyastha, “Spanstore: Cost-effective geo-replicated storage

spanning multiple cloud services," in *ACM Symposium on Operating Systems Principles*, 2013, pp. 292–308.

[11] O. Goldreich, "Towards a theory of software protection and simulation by oblivious rams," in *ACM symposium on Theory of computing*. ACM, 1987, pp. 182–194

[12] R. Ostrovsky, "Efficient computation on oblivious rams," in *Proceedings of ACM STOC*, 1990, pp. 514–523.

[13] O. Goldreich and R. Ostrovsky, "Software protection and simulation on oblivious rams," *Journal of the ACM (JACM)*, vol. 43, no. 3, pp. 431–473, 1996.

[14] E. Kushilevitz, S. Lu, and R. Ostrovsky, "On the (in) security of hashbased oblivious ram and a new balancing scheme," in *ACM-SIAM SODA*, 2012, pp. 143–156.

[15] P. Williams, R. Sion, and B. Carbunar, "Building castles out of mud: practical access pattern privacy and correctness on untrusted storage," in *Proceedings of the 15th ACM conference on Computer and communications security*, 2008, pp. 139–148.

[16] P. Williams and R. Sion, "Single round access privacy on outsourced storage," in *ACM CCS*, 2012, pp. 293–304.

[17] J. R. Lorch, B. Parno, J. Mickens, M. Raykova, and J. Schiffman,"Shroud: ensuring private access to large-scale data in the data centers," in *USENIX FAST*, 2013, pp. 199–213.