Optimal Performance and Security in Division and Replication of Data in Cloud-OPoR

B.Preethi,A.Jeya priya,S.Priya,A.Bhanumathi,
UG Student,
Dept of Computer Science and Engg.
Mother Terasa College of Engineering and Technology,
Anna University Chennai,
Pudukkottai, India.
Preethi17green@gmail.com

R.Vidthya,ME
Assistant Professor,
Dept of Computer Science and Engg.
Mother Terasa College of Engineering and Technology,
Anna University Chennai,
Pudukkottai, India.
vidthyahicet@gmail.com

***Abstract*— Cloud Computing moves the application software and databases to the centralized large data centers, where the management of the data and services may not be fully trustworthy. Here to study the problem of ensuring the integrity of data storage in Cloud Computing .To reduce the computational cost at user side during the integrity verification of their data, and to tackle the challenges. OPoR is introducing a new cloud storage scheme involve a cloud storage server and a cloud audit server. For improving the strengthen Proof of Retrievabiliy (PoR) model to support dynamic data operations. Here processing data as fragmentation of files and replicated data. Files are assigned in a no of creating nodes and to be stored on the cloud server. When the user downloading the file from the server then it retrieve from those nodes and automatically merge those files and download at destination**.

I. INTRODUCTION

Cloud Computing has been envisioned as the next generation architecture of the IT enterprise due to its long list of unprecedented advantages: on-demand self service, ubiquitous network access, location-independent resource pooling, rapid resource elasticity, and usage based pricing. In particular, the ever cheaper and more powerful processors, together with the "software as a service" (SaaS) computing architecture, are transforming data centers into pools of computing service on a huge scale. Although having appealing advantages as a promising service platform for the Internet, this new data storage paradigm in "Cloud" brings many challenging issues which have profound influence on the usability, reliability, scalability, security, and performance of the overall system. One of the biggest concerns with remote data storage is that of data integrity verification at untrusted servers. For instance, the storage service provider may decide to hide such data loss incidents as the Byzantine failure from the clients to maintain a reputation. What is more serious is that for saving money and storage space the service provider might deliberately discard rarely accessed data files which belong to an ordinary client. Considering the large size of the outsourced electronic data and the client's constrained resource capability, the core of the problem can be generalized as how can the client find an efficient way to perform periodical integrity verification without the local copy of data files. In order to overcome this problem, many schemes have been proposed under different system and security models [1]–[10]. In all these works, great efforts have been made to design solutions that meet various requirements: high scheme efficiency, stateless verification, unbounded use of queries and retrievability of data, etc. According to the role of the verifier in the model, all the schemes available fall into two categories: private verifiability and public verifiability. Although achieving higher efficiency, schemes with private verifiability impose computational burden on clients. On the other hand, public verifiability alleviates clients from performing a lot of computation for ensuring the integrity of data storage. To be specific, clients are able to delegate a third party to perform the verification without devotion of their computation resources. In the cloud, the clients may crash unexpectedly or cannot afford the overload of frequent integrity checks. Thus, it seems more rational and practical to equip the verification protocol with public verifiability, which is expected to play a more important role in achieving better efficiency for Cloud Computing. What's more, there is another major concern among previous designs, that is the support of dynamic data operation for cloud data storage applications. In Cloud Computing, the remotely stored electronic data might not only be accessed but also be updated by the clients, e.g., through block modification, deletion, insertion etc. Unfortunately, the-state-of-the-art in the context of remote data storage mainly focus on static data files and this dynamic data updates has received limited attention in the data possession applications so far [1]–[3], [9], [11]. Though such problem also has been addressed in [12]. it is well believed that supporting dynamic data operation can be of vital importance to the practical application of storage-outsourcing services. In view of the key role of public verifiability and dynamic data operation support for cloud data storage, in this paper we present a framework and an efficient construction for seamless integration of these two components in our protocol

design. In addition, most of existing works adopt weaker security models which do not take into account the reset attack. Specifically, the cloud storage server can trigger reset attacks in the upload phase to violate the soundness of the scheme. To the best of our knowledge, it seems that no existing scheme can simultaneously provide provable security in the enhanced security model and enjoy desirable efficiency, that is, no scheme can resist reset attacks while supporting efficient public verifiability and dynamic data operations simultaneously.

Contributions: Our contribution can be summarized as follows:

• We propose OPoR, a new PoR scheme with two independent cloud servers. Particularly, one server is for auditing and the other for storage of data. The cloud audit server is not required to have high storage capacity. Different from the previous work with auditing server and storage server, the user is relieved from the computation of the tags for files, which is moved and outsourced to the cloud audit server. Furthermore, the cloud audit server also plays the role of auditing for the files remotely stored in the cloud storage server.
• We develop a strengthened security model by considering the reset attack against the storage server in the upload phase of an integrity verification scheme. It is the first PoR model that takes reset attack into ccount for cloud storage system.
• We present an efficient verification scheme for ensuring remote data integrity in cloud storage. The proposed scheme is proved secure against reset attacks in the strengthened security model while supporting efficient public verifiability and dynamic data operations simultaneously

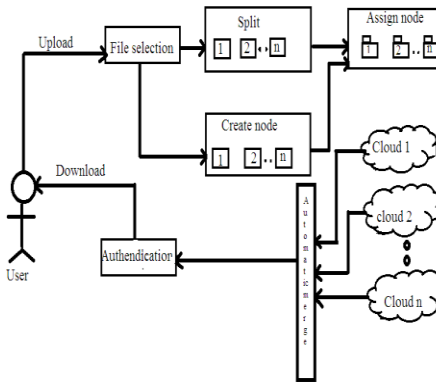

Figure 1.

Figure 2. Figure 1. Architecture Diagram

II. RELATED WORK

Recently, much research effort has been devoted largely to ensure the security of cloud computing [1] and remotely stored data [1]–[3], [9], [11]. Ateniese *et al.* [1] defined the "provable data possession" (PDP) model for ensuring possession of files on untrusted storages. They also proposed the first proofof- storage scheme that supports public verifiability. The scheme utilizes RSA-based homomorphic tags for auditing outsourced data, such that a linear combination of file blocks can be aggregated into a single block and verified by employing homomorphic property of RSA. However, the data owner has to compute a large number of tags for those data to be outsourced, which usually involves exponentiation and multiplication operations. Furthermore, The case of dynamic data storage has not been considered by Ateniese *et al.*, and the direct extension of their scheme from static data storage to dynamic case brings many security problems. In their subsequent work [11], Ateniese *et al.* proposed a dynamic version of the prior PDP scheme. However, the system imposes a priori bound on the number of queries and do not support fully dynamic data operations. In considered dynamic data storage in distributed scenario, and the proposed challenge-response protocol can both determine the data correctness and locate possible errors. Similar to [11], they only considered partial support for dynamic data operation. In they also considered how to save storage space by introducing deduplication in cloud storage. Recently, introduced the provable data possession problem in a cooperative cloud service providers and designed a new remote integrity checking system. Juels *et al.* [2] introduced a "proof of retrievability" (PoR) model, where spot-checking and error-correcting codes are adopted to ensure both "possession" and "retrievability" of data files in archive service systems. However, public verifiability is not supported in their scheme and the data owner also has to make many computational efforts to generate tags for those data to be outsourced. Shacham *et al.* [3] designed an improved PoR scheme with public verifiability based on BLS signature and the proofs are given in a stronger security model defined in [2]. Similar to the construction in [1], they used publicly verifiable homomorphic authenticators that are built from BLS signatures and proven secure in the random oracle model. For the first time, explored constructions for dynamic provable data possession. They extended the PDP model in [1] to support provable updates to stored data files using rank-based authenticated skip lists. This scheme is essentially a fully dynamic version of the PDP solution. In particular, to support updates, especially for block insertion, they particular data or to search most relevant data. It is very challenging for search engine to fetch relevant data as per user's need and which consumes

more time. So, to reduce large amount of time spend on searching most relevant data we proposed the "Advanced crawler". In this proposed approach, results collected from different web search engines to achieve Meta search approach. Multiple search engine for the user query and aggregate those result in one single space and then performing two stages crawling on that data or Urls. In which the sight locating and in-site exploring is done f or achieving most relevant site with the help of page ranking and reverse searching techniques. This system also works online and offline manner.

I. III. DESIGN

a. PROFILE GENERATION

This module is login for existing users and registration for new or upcoming users. When the user has register the details and after the details are stored in the cloud server or database. Then after the given user has assigned Username and Password for login the webpage. Then after the user has enter the username & password is correct and then only to go for specified page.

b. CLIENT-SERVER COMMUNICATION

Normally, a server runs on a specific computer and has a socket that is bound to a specific port number. The server just waits, listening to the socket for a client to make a connection request. On the client-side: The client knows the hostname of the machine on which the server is running and the port number on which the server is listening. To make a connection request, the client tries to rendezvous with the server on the server's machine and port. The client also needs to identify itself to the server so it binds to a local port number that it will use during this connection. This is usually assigned by the system. The client is sending request to the server for processing or giving some needs.

c. FILE SPLITTING BY DATA OWNER

The owner can best split the file such that each fragment does not contain significant amount of information as the owner is cognizant of all the facts pertaining to the data. Once the file is split into fragments, the DROPS methodology selects the cloud nodes for fragment placement. The selection is made by keeping an equal focus on both security and performance in terms of the access time.

D. CREATION AND SELECTION OF NODE

In the second iteration that node is selected that produces the lowest RC in combination with node already selected. The process is repeated for all of the file fragments. The centrality measure is the same for all of the nodes. This results in the selection of same node for storing the file fragment. Consequently, the performance showed the same value and all three lines are on the same points. However, this is not the case for the Dcell architecture.

E. FILE MERGING

To placing the fragments on the central nodes, the system perform a controlled replication to increase the data availability, reliability, and improve data retrieval time. The system place the fragment on the node that provides the decreased access cost with an objective to improve retrieval time for accessing the fragments for reconstruction of original file. While replicating the fragment, the separation of fragments as explained in the placement technique through T-coloring, is also taken care off. It is also possible that some of the fragments are left without being replicated because of the T-coloring.

F. PERFORMANCE ANALYSIS

The behavior of the algorithms was studied by Increasing the number of nodes in the system, Increasing the number of objects keeping number of nodes constant, Changing the nodes storage capacity, and Varying the read/write ratio. The aforesaid parameters are significant as they affect the problem size and the performance of algorithms.

FEASIBILITY SYSTEM

We analyze the security of our scheme under a variant of Shacham and Waters' PoR model . which supports public verifiability and dynamic update operations Besides, our model offers strengthened security by allowing a alicious storage server to perform a reset attack against the client and the cloud audit server in upload phase. The basic goal of PoR model is to achieve proof of retrievability. Informally, this property ensures that if an adversary can generate valid integrity proofs of any file *F* for a non-negligible fraction of challenges, we can construct a PPT machine to extract *F* with overwhelming probability.

- Technical Feasibility
- Operational Feasibility
- Economical Feasibility
- Schedule Feasibility

IV. METHODOLOGY

Suppose we have a graph G = (V, E) and a set T containing non-negative integers including 0. The Tcoloring is a mapping function f from the vertices of V to the set of non-negative integers, such that $|f(x)- f(y)| \notin T$, where $(x, y) \in E$. The mapping function f assigns a color to a vertex. In simple words, the distance between the colors of the adjacent vertices must not belong to T.

Formulated by Hale [6], the T-coloring problem for channel assignment assigns channels to the nodes, such that the channels are separated by a distance to avoid interference.

V. CONCLUSION

In this paper, Here proposes OPoR, a new proof of retrievability for cloud storage, in which a trustworthy audit server is introduced to pre-process and upload the data on behalf of the clients. In OPoR, the computation overhead for tag generation on the client side is reduced significantly. Besides, we construct another new PoR scheme proven secure under a PoR model with enhanced security against reset attack in the upload phase. There are several interesting topics to do along this research line.

[1]

[2]

[3]

[4]

[5]

References

[1] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, "Provable data possession at untrusted stores," in *CCS '07: Proceedings of the 14th ACM conference on Computer and communications security*. New York, NY, USA ACM, 2007, pp. 598–609.

[2] A. Juels and B. S. K. Jr., "Pors: proofs of retrievability for arge files," in *CCS '07: Proceedings of the 14th ACM conference on Computer and communications security*. New York, NY, USA: ACM, 2007, pp. 584–597.

[3] H. Shacham and B. Waters, "Compact proofs of etrievability," in *ASIACRYPT '08: Proceedings of the14thInternationalConference on the Theory and Application of Cryptology and Information Security*. Berlin, Heidelberg: Springer-Verlag, 2008, pp. 90–107.

[4] K. D. Bowers, A. Juels, and A. Oprea, "Proofs of retrievability: theory and implementation," in *Proceedings of CCSW 2009*. ACM, 2009, pp. 43–54.

[5] M. Naor and G. N. Rothblum, "The complexity of online memory checking," *J. ACM*, vol. 56, no. 1, pp. 2:1–2:46, Feb 2009. [Online]. Available: http://doi.acm.org/10.1145/1462153. 1462155

[6] E.-C. Chang and J. Xu, "Remote integrity check with dishonest storage server," in *Proceedings of ESORICS 2008, volume 5283 of LNCS*. Springer-Verlag, 2008, pp. 223–237.

[7] M. A. Shah, R. Swaminathan, and M. Baker, "Privacy-preserving audit and extraction of digital contents," Cryptology ePrint Archive, Report 2008/186, 2008, http://eprint.iacr.org/.

[8] A. Oprea, M. K. Reiter, and K. Yang, "Space-efficient block storage integrity," in *In Proc. of NDSS 2005*, 2005.

[9] T. S. J. Schwarz and E. L. Miller, "Store, forget, and check: Using algebraic signatures to check remotely administered storage," in *ICDCS '06: Proceedings of the 26th IEEE International Conference on Distributed Computing Systems*. Washington, DC, USA: IEEE Computer Society, 2006.

[10] Q. Wang, K. Ren, S. Yu, and W. Lou, "Dependable and secure sensor data storage with dynamic integrity assurance," *ACM ansactions on Sensor Networks*, vol. 8 no. 1, pp. 9:1–9:24, Aug. 2011. [Online]. Available: http: //doi.acm.org/10.1145/1993042.1993051

[11] L. V. M. Giuseppe Ateniese, Roberto Di Pietro and G. Tsudik, "Scalable and efficient provable data possession," in *International Conference on Security and Privacy in Communication Networks (SecureComm 2008)*, 2008.

[12] C. Wang, Q. Wang, K. Ren, and W. Lou, "Privacy-preserving public auditing for data storage security in cloud computing," in I*NFOCOM*, 2010, pp. 525–533.

[13] Y. Zhu, H. Wang, Z. Hu, G.-J. Ahn, H. Hu, and S. S. Yau, "Dynamic audit services for integrity verification of outsourced storages in clouds," in *SAC*, 2011, pp. 1550–1557.

[14] Q. Zheng and S. Xu, "Fair and dynamic proofs of retrievability," in *CODASPY*, 2011, pp. 237–248.

[15] J. Li, X. Chen, J. Li, C. Jia, J. Ma, and W. Lou, "Fine-grained access control system based on attribute-based encryption," *ESORICS*, 2013.

[16] A.Suresh (2016), "Evaluation of Quality of Service through Genetic Approach in Telecommunication", in International Journal of Control Theory and Applications, (IJCTA) ISSN: 0974-5572, *Vol. 09, No.36, December 2016*, pp.*409* – 417.