

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 9, Issue 3, March 2020

Identity Based Dynamic Data Auditing and Recovery Using Cloud Computing

Mohanadevi MC

U.G Student, Department of Computer Science and Engineering, Loyola ICAM College of Engineering and
Technology, Chennai, Tamil Nadu, India

ABSTRACT: Auditing is a process of verification of customer information which can be carried out either by the customer himself or by a TPA (Third Party Auditor/Trusted Party Auditing). A **Data audit** refers to the auditing of data to assess its quality or utility for a specific purpose. Auditing data, unlike auditing finances, involves looking at key metrics, other than quantity, to create conclusions about the properties of a data set. The objective of the project is to audit the data that we upload on the cloud and split those data into chunks. Finally using TPA we will verify the data whether any malfunction was on data. After that we use the concept of ERASURE CODE to recover the files from the replica. We are deploying this Application in Cloud. Data is made & Encrypted, Split and stored in Cloud. Replica is created for data backup. Top Hash Key is stored in Separate Cloud as well in the Local Backup. We apply the MHT algorithm for data splitting. In our implementation we are using cloud for data storage. Because in the existing system automatic data recovery is drawback. If a user loses any file in their cloud storage, the user has to give a recovery option and the cloud service provider will recover the file and provide that to the user. But in this paper we are implementing a system that will recover the crashed files automatically.

KEYWORDS: TPA, Auditing, MHT, Hash key, Erasure code, Replica

I. INTRODUCTION

Cloud computing is a fast-growing technology that has established itself in the next generation of IT industry and business. Cloud computing promises reliable software, hardware, and IaaS delivered over the Internet and remote data centers. The concept of **Data auditing** is introduced in Cloud Computing to deal with secure information storage. Auditing is a process of verification of customer information which can be carried out either by the customer himself or by a TPA (Third Party Auditor/Trusted Party Auditing). The existing remote data auditing schemes mainly use public key infrastructure (PKI) and identity-based cryptography. However, PKI has the drawback of overspending on key management, so identity-based cryptography is more commonly used. After analyzing the existing schemes, we found that there is none of identity-based remote data auditing schemes which can support dynamic auditing. Moreover, these schemes cannot be extended to dynamic auditing schemes directly. In these schemes, tag generation is linked to the index of the data block. For example, if a data block m_2 is inserted after m_1 , for each block whose index is behind, its tag has to be recomputed, which leads to large computation cost. So these schemes cannot be extended to dynamic auditing schemes directly. We propose an identity-based dynamic data auditing scheme for big data storage.

The main contributions of our work are:

(1) We design an identity-based dynamic data auditing scheme that is capable of performing dynamic auditing for big data storage. Dynamic data auditing is an important feature in current identity-based remote data auditing schemes. With proper algorithms to generate block tags based on bilinear maps, our scheme first achieves dynamic data auditing in identity based remote data auditing schemes.

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 9, Issue 3, March 2020

(2) To guarantee the data is updated correctly each time, we use a data structure, namely Merkle hash tree (MHT), to authenticate block tags and support dynamic operation with integrity assurance. With this data

structure, our scheme efficiently supports dynamic data operations, including modification, insertion and deletion.

(3) We have TPA Third Party Auditing to monitor the file crashes TPA will intimate the user that the filer crashed . After that we are using the concept Erasure code , it will recover the file from the replica server. So through this we are providing security to the user and his data.

II. RELATED WORKS

Huayu Zhang: In order to guarantee data reliability in distributed storage systems, erasure codes are widely used for the desirable storage properties. Nevertheless, the codes have one drawback that overmuch data is needed to repair a failure, resulting in both large bandwidth consuming in the network and high calculation pressure on the replacement node. For repair bandwidth problems, researchers derive the tradeoff between storage and repair traffic from network coding and propose regenerating codes. However, the constructions of regenerating codes complicate the systems as well as recovery calculation. Hence, this paper proposes a distributed repair method based on general erasure codes to mitigate the burden of both recovery computation and network traffic. We observe that distributing recovery computation among helpers can distract the whole calculation procedure and accelerate repair speed in practical systems. Furthermore, by combining this technique with network topology, we introduce a novel repair tree to minimize repair traffic. Repair tree is also derived from network coding. The performance of the repair tree is preliminarily analyzed and evaluated, which infers that the storage-bandwidth bound of regenerating codes can be broken under this model[1].

Alexandros G. Dimakis: Distributed storage systems provide reliable access to data through redundancy spread over individually unreliable nodes. Application scenarios include data centers, peer-to-peer storage systems, and storage in wireless networks. Storing data using an erasure code, in fragments spread across nodes, requires less redundancy than simple replication for the same level of reliability. However, since fragments must be periodically replaced as nodes fail, a key question is how to generate encoded fragments in a distributed way while transferring as little data as possible across the network. For an erasure coded system, a common practice to repair from a single node failure is for a new node to reconstruct the whole encoded data object to generate just one encoded block. We show that this procedure is sub-optimal[2].

Hussein F. Salama: Multicast (MC) routing algorithms capable of satisfying the quality of service (QoS) requirements of real-time applications will be essential for future high-speed networks. We compare the performance of all of the important MC routing algorithms when applied to networks with asymmetric link loads. Each algorithm is judged based on the quality of the MC trees it generates and its efficiency in managing the network resources. Simulation results over random networks show that unconstrained algorithms are not capable of fulfilling the QoS requirements of real-time applications in wide-area networks. Simulations also reveal that one of the unconstrained algorithms, reverse path multicasting (RPM), is quite inefficient when applied to asymmetric networks. We study how combining routing with resource reservation and admission control improves RPM's efficiency in managing the network resources. The performance of one semiconstrained heuristic, MSC, three constrained Steiner tree (CST) heuristics, Kompella, Pasquale, and Polyzos (KPP), constrained adaptive ordering (CAO), and bounded shortest multicast algorithm (BSMA), and one constrained shortest path tree (CSPT) heuristic, the constrained Dijkstra heuristic (CDKS) are also studied. Simulations show that the semiconstrained and constrained heuristics are capable of successfully constructing MC trees which satisfy the QoS requirements of real-time traffic. However, the cost performance of the heuristics varies. BSMA's MC trees are lower in cost than all other constrained heuristics. Finally, we compare the execution times of all algorithms, unconstrained, semiconstrained, and constrained[3].

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 9, Issue 3, March 2020

Ling Liu: With fast paced growth of digital data and exploding storage management costs, enterprises are looking for new ways to effectively manage their data. One such cost-effective paradigm is the Storage-as-a-Service model, in which enterprises outsource their storage to a storage service provider (SSP) by storing data at a remote SSP-managed site and accessing it over a high speed network. Often for a variety of reasons, enterprises find it unacceptable to fully trust the SSP and prefer to store data in an encrypted form. This typically limits collaboration and data sharing among enterprise users due to complex key management and access control challenges. In this paper, we propose a platform called SHAROES that provides data sharing capability over such outsourced storage environments. SHAROES provide rich *nix-like data sharing semantics over SSP stored data, without trusting the SSP for data confidentiality or access control. SHAROES is unique in its ability in reducing user involvement during setup and operation through the use of in-band key management and allows a near-seamless transition of existing storage environments to the new model. It is also superior in performance by minimizing the use of expensive public-key cryptography in metadata management. We present the architecture and implementation of various SHAROES components and our experiments demonstrate performance superior to other proposals by over 40% on a number of benchmarks[4].

Muhammad Khurram Khan : Cloud computing has emerged as a computational paradigm and an alternative to conventional computing with the aim of providing reliable, resilient infrastructure, and with high quality of services for cloud users in both academic and business environments. However, the outsourced data in the cloud and the computation results are not always trustworthy because of the lack of physical possession and control over the data for data owners as a result of using virtualization, replication and migration techniques. Since the security protection the threats to outsourced data have become a very challenging and potentially formidable task in cloud computing, many researchers have focused on ameliorating this problem and enabling public auditability for cloud data storage security using remote data auditing (RDA) techniques. This paper presents a comprehensive survey on the remote data storage auditing in single cloud server domain and presents taxonomy of RDA approaches. The objective of this paper is to highlight issues and challenges to current RDA protocols in the cloud and mobile cloud computing. We discuss the thematic taxonomy of RDA based on significant parameters such as security requirements, security metrics, security level, auditing mode, and update mode. The state-of-the-art RDA approaches that have not received much coverage in the literature are also critically analyzed and classified into three groups of provable data possession, proof of retrievability, and proof of ownership to present a taxonomy. It also investigates similarities and differences in such frameworks and discusses open research issues as the future directions in RDA research[5].

III. PROPOSED SYSTEM

We are deploying this Application in Cloud. Data is made & Encrypted, Split and stored in Cloud. Replica is created for data backup. Top Hash Key is stored in Separate Cloud as well in the Local Backup. We apply the MHT algorithm for data splitting. In our implementation we are using cloud for data storage . Because in the existing system automatic data recovery is a drawback.

If a user loses any file in their cloud storage user has to give a recovery option and the cloud service provider will recover the file and provide that to the user. But in this paper we are implementing a system that will recover the crashed files automatically.

Based on the Data owner request , during the upload session the requested file is uploaded to the cloud storage after the data is encrypted then those data move onto the hashing process after the data get hashed then they are all splitted into Eight parts using MHT algorithm then the datas are stored into the CLOUD SERVER. Once the UPLOAD is done then we can audit the uploaded file if needed . If the file gets corrupted by anyone then the TPA will Intimate the users through the Email.

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 9, Issue 3, March 2020

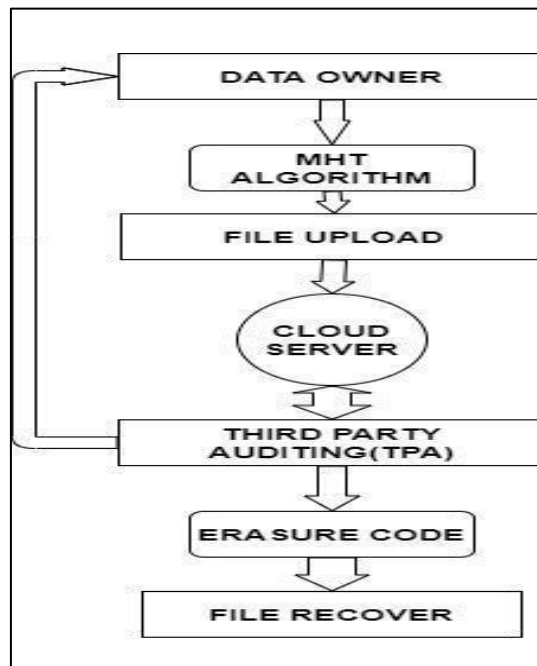


Fig.1 The architecture of the system

MHT ALGORITHM

- In this module, Once the data is uploaded the entire data is chunked into multiple parts using the Hash Tree algorithm and stored in separate data servers.
- In cryptography and computer science a hash tree or Merkle tree is a tree in which every leaf node is labeled with the hash of a data block and every non-leaf node is labeled with the cryptographic hash of the labels of its child nodes. Hash trees allow efficient and secure verification of the contents of large data structures. Hash trees are a generalization of hash and hash chains.

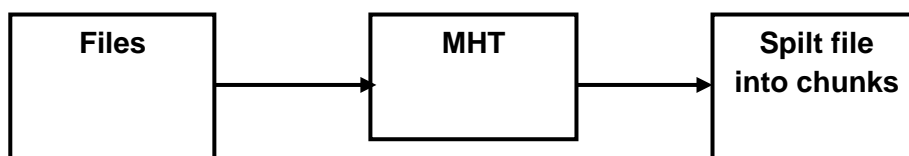


Fig.2 MHT Function

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 9, Issue 3, March 2020

ERASURE CODE

- ❑ This is the main module of the project, if data is lost then data can be retrieved from the Replica server.
- ❑ Assuming that data is lost in the replica server also then the data has to be reconstructed using erasure code implementation.
- ❑ Data is Split into 2 parts like D1234 & D5678, and then these two are Split into 4 parts like D12, D34, D56, D78. It is further Split into 8 parts like D1, D2, D3, D4, D5, D6, D7, D8 and stored in two different Cloud separate servers.
- ❑ The main Hash value of D12345678 is called Top Hash Code which is stored separately.
- ❑ IF D1 is lost from the data server and also from the replica server then Top hash is considered and subtracted it from D5678, we will get D1234, which is again subtracted from D34 and we will get D12, which is again subtracted from D2 and we will get D1.

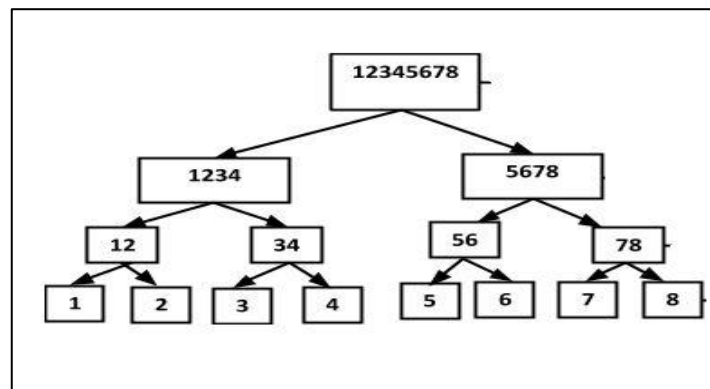


Fig.3 MHT process

THIRD/TRUSTED PARTY AUDITING

- ❑ Once the parity adds bits, then the data will be given to the Trusted Parity auditor.
- ❑ The Trusted Parity Auditor will generate the signature using change and response methods.
- ❑ The data will be audited in this module, if any changes occur it will provide the intimation regarding the changes to the required user.

UPLOAD FILE

In this Module User interface is created so that the data owner will upload the data to the server. The main objective of this module is to store and share the data by uploading the file to the remote machine. Data is Hashed and applied XOR functionality and then finally stored in the main server. Data owners will upload their data to the cloud server and a request for a particular file is sent to the cloud server. Both the upload and the file request are handled by the main Cloud Server. During the file request is processed the main server will communicate with the data owner and the files are retrieved only after the approval given the data owner.

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 9, Issue 3, March 2020

REPLICA STORAGE

Once the data is stored in the corresponding data servers and the keys are stored in the key servers. Parity bit is added to the data to make the data so secured. Then we're adding the parity bits to the data, so that the data will be changed. These chunked data is stored in the other servers as replicas so as to retrieve the data when the main storage data is lost.

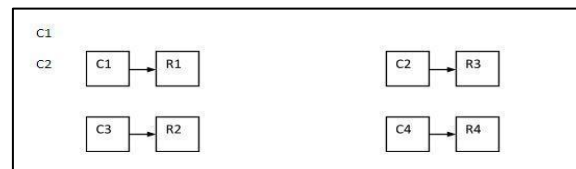


Fig.4 Replica of the Files

FILE DOWNLOAD

The Data owner will download the file from the cloud while the user downloads all separated files and will be reconstructed into a single file and stored on local memory. While file download from the cloud the binary bits files are converted into text files and view those to the user. For upload and download the file, code should be configured with cloud API using App key, secret key and token of the user's cloud account.

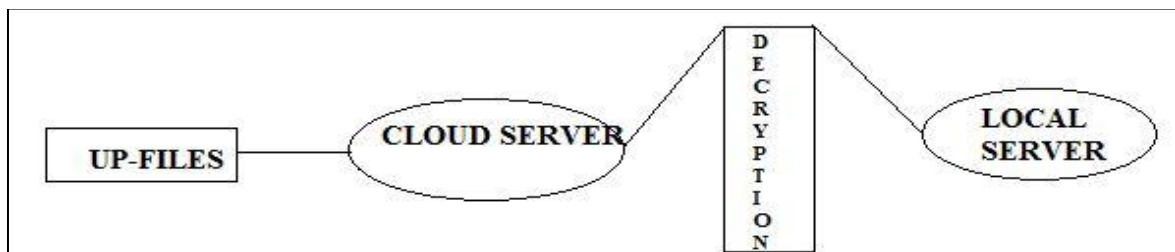


Fig.5 Download Process

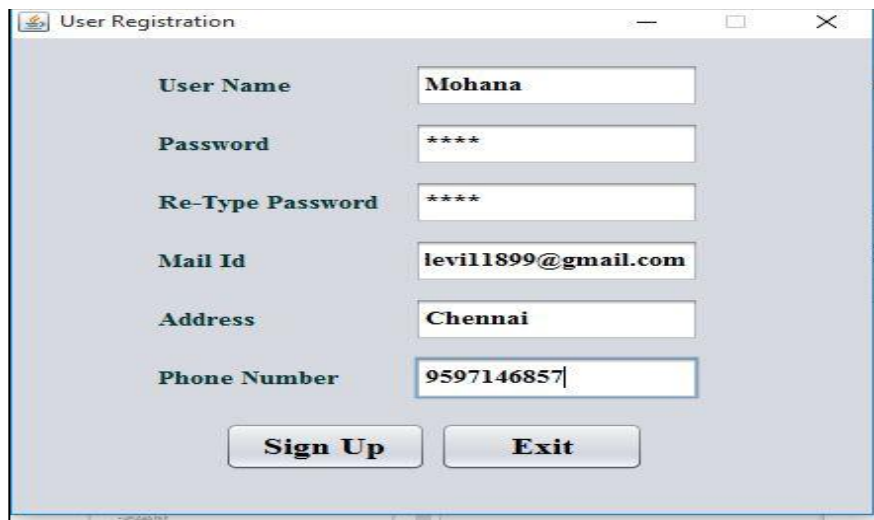
International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 9, Issue 3, March 2020

IV. RESULTS



A screenshot of a web browser window titled "User Registration". The form contains the following fields and values:

User Name	Mohana
Password	****
Re-Type Password	****
Mail Id	levil1899@gmail.com
Address	Chennai
Phone Number	9597146857

At the bottom of the form are two buttons: "Sign Up" and "Exit".

Fig.6 User login page

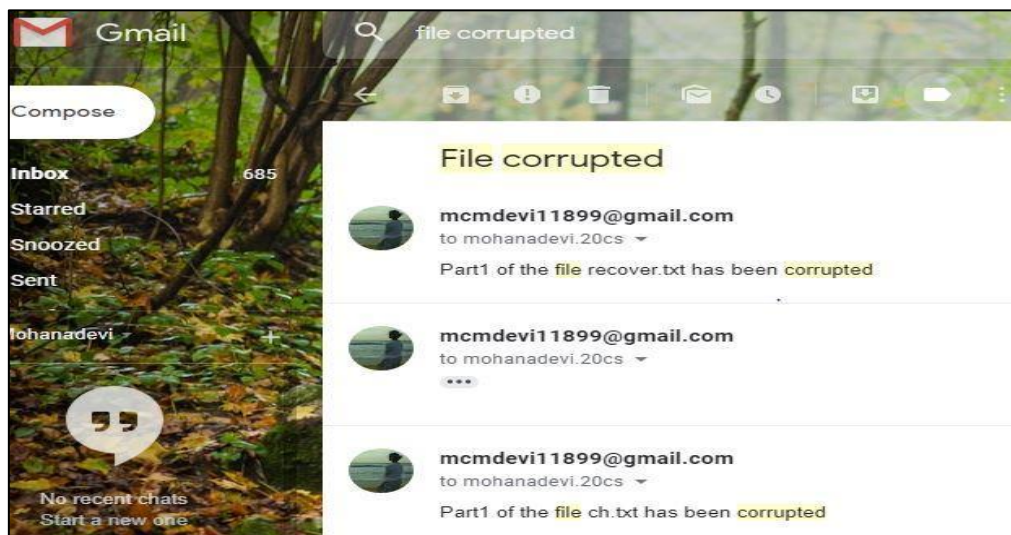


Fig.7 Alert message

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 9, Issue 3, March 2020



Fig.7 Corrupted File Download

V. CONCLUSION

Thus the paper infer that nowadays people are using the cloud in a huge number of ways, but the question is whether proper security is there or not? We aren't able to provide security to the entire cloud server but we are able to secure our data. Here, we provide the security system using MHT by splitting the data on cloud server and if in any case our data was crashed on cloud TPA will send notification to the user through Mail and also the crashed file will automatically be recovered using Replica server. The future enhancement of this paper is we can secure the file by uploading the data on two different clouds like google Drive and Dropbox . When the user downloads the file it will reconstruct the data from those clouds. By this way unauthorized users cannot view or edit the file on the cloud.

VI. ACKNOWLEDGEMENT

This work is a part of project development supported by the Computer Science department of Loyola ICAM college of Engineering and Technology (2020).

REFERENCES

- [1] M. Sookhak, A. Gani, H. Talebian, S. U. Khan, R. Buyya, and A. Y. Zomaya, "Remote data auditing in cloud computing environments: A survey, taxonomy, and open issues," *ACM Comput. Surveys*, vol. 47, no. 4, pp. 65:1–65:34, May 2015.
- [2] A. Singh and L. Liu, "Sharoes: A data sharing platform for outsourced enterprise storage environments," in *Proc. IEEE 24th Int. Conf. Data Eng.*, Apr. 2008, pp. 993–1002.
- [3] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, "Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility," *Future Generation Computer. Syst.*, vol. 25, no. 6, pp. 599–616, 2009.
- [4] C. Wang, K. Ren, W. J. Lou, and J. Li, "Toward publicly auditable secure cloud data storage services," *IEEE Netw.*, vol. 24, no. 4, pp. 19–24, Jul./Aug. 2010.
- [5] L. Wei, et al., "Security and privacy for storage and computation in cloud computing," *Inf. Sci.*, vol. 258, pp. 371–386, Feb. 2014.
- [6] G. Ateniese, R. D. Pietro, L. V. Mancini, and G. Tsudik, "Scalable and efficient provable data possession," in *Proc. ACM Int. Conf. Secur. Privacy Commun. Netw.*, 2008, pp. 9:1–9:10.
- [7] L. Chen, "Using algebraic signatures to check data possession in cloud storage," *Future Generation Comput. Syst.*, vol. 29, no. 7, pp. 1709–1715, Sep. 2013.
- [8] C. Liu, et al., "Authorized public auditing of dynamic big data storage on cloud with efficient verifiable fine-grained updates," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 9, pp. 2234–2244, Sep. 2013.
- [9] K. Yang and X. Jia, "Data storage auditing service in cloud computing: Challenges, methods and opportunities," *World Wide Web*, vol. 15, no. 4, pp. 409–428, 2011.
- [10] B. Chen, R. Curtmola, G. Ateniese, and R. Burns, "Remote data checking for network coding-based distributed storage systems," in *Proc. ACM Workshop Cloud Comput. Secur. Workshop*, 2010, pp. 31–42.
- [11] G. Ateniese, et al., "Provable data possession at untrusted stores," in *Proc. 14th ACM Conf. Comput. Commun. Secur.*, 2007, pp. 598–609.
- [12] R. Curtmola, O. Khan, and R. Burns, "Robust remote data checking," in *Proc. 4th ACM Int. Workshop Storage Secure. Survivability*, 2008, pp. 63–68.