Secure Dynamic Data Support and Trusted Third Party Auditor In Cloud Computing

Mrs. O. Rajitha **Dr.S. Murali Krishna Professor & Head**

ABSTRACT:

The cloud storage is one which enables users to store their data remotely and enjoy on-demand quality cloud services without the burden of local software and hardware management. So benefits are clear to us, that type of service is also relinquishing physical possession of user's outsourced data, which inevitably poses the new security risks towards correctness of the user's data in the cloud. To address this problem and further achieve a secure and dynamic data support and trusted auditing mechanism in for cloud storage service, here in this paper we propose a flexible distributed storage integrity third party auditing mechanism, by using the token keys and privacy preserved sharing of keys to their mails. Our proposed design allows the user or third party auditor to audit the cloud storage with low computation cost and very light weight communication.The Third party user is also one cloud server. The auditing result not only achieves the fast data error localization(identifying the misbehaving server) but also ensures strong cloud storage correctness guarantee. By considering cloud data is dynamic in nature, the proposed design further supports efficeint and secure dynamic operations on outsourced data, which includes block modification, append, and deletion. The above analysis shows that the proposed scheme is highly efficient and resilient against Byzantine failure,server colluding attacks and malicious data modification attack.

Index terms: Cloud computing, error localization, data dynamics, storage integrity, auditing mechanism,token keys,against byzantine failure.

1. INTRODUCTION

Storing data into the cloud offers the great convenience to the users since they don't have to take care about the complexities of direct software and hardware management. The best cloud service providers are Amazon Simple Storage Service (S3) and Amazon Elastic Compute Cloud(EC2) are both well known examples.Although the cloud infrastructures are much more powerful and reliable than personal computing devices.In this paper, we propose an effective and flexible distributed scheme with explicit dynamic data support to ensure the correctness of users' data in the cloud through third party auditor. In olden days we depends on erasure correcting code in the file distribution preparation to provide redundant copies and guarantees dependability of data. This may causes slowly reduces the communication and storage overhead problems as compared to the old replication-based file distribution techniques. By using the token keys with distributed verification of erasure-coded data, our scheme achieves the correctness of strores data in the cloud as well as data error localization: whenever data corruption has been detected during the storage correctness verification, our scheme can almost guarantee the simultaneous localization of data errors, i.e., the identification of the misbehaving server(s). Frequently, when we mention the cloud computing issue, enormous threats are raised .One of the major threats are data privacy and integrity.

A lot of research discuss this problem and introduce many solutions to decrease the threat of the data privacy and integrity. Somebody introduce threat model to treat the privacy problem in the clouds. One of the threats in cloud computing is tampering with data in the cloud that interfere with the unauthorized modifications for the data, which lead to an effectiveness on processors, data storage and data flow. Then, they suggested different solutions technique for this threat. One of the solutions is using digital signature which will be used in their model.In our paper we introduce token key computation and encryption techniques which follows Rijandael algorithm to provide the security. And by using erasure coded data is used to check storage integrity.

Some techniques can be useful to ensure the storage correctness without having users involvement to check local data, they are all focusing on single server scenario. They may be useful for quality-of-service testing , but does not guarantee the data availability in case of server failures. So to overcome this problem we are using multi cloud technology which stores users data in distributed multiple servers as resulted storage verification overhead would be linear to the number of servers. How- ever, while providing efficient cross server storage verification and data availability insurance, these schemes are all focusing on static or archival data. As a result, their capabilities of handling dynamic data remains unclear, which inevitably limits their full applicability in cloud storage scenarios.

Our work is among the first few ones in this field to consider distributed data storage in Cloud Computing. Our contribution can be explained as the following three things:

1) Compared to many of its previous systems, they only provide binary results about the storage state across the distributed servers, to localize error data the challenge-response protocol is used .

2) Unlike most predecessors no need to ensure data integrity remotely, our new scheme supports secure and efficient dynamic operations on the data blocks which includes: update, delete and append operations which will be performed on the cloud data.

3) Extensive security and performance analysis shows that the proposed scheme is highly efficient and resilient against Byzantine failure, malicious data modification attack, and even server colluding attacks.

In order to overcome the user's risks towards verifying the data in the cloud, the user must be able to use the assistance of a Third Party Auditor (TPA). The TPA has an experience that clouds users does not have, and checks over to verify storage correctness that is difficult for the users to check. The auditing result not only achieves the fast data error localization (identifying the misbehaving server) but also ensures strong cloud storage correctness guarantee.

2 RELATED WORK:

2.1 PDP scheme:

Cloud Computing environment is constructed based on open architectures and interfaces; it has the capability to incorporate multiple internal and/or external cloud services together to provide high interoperability. By using Provable Data Possession (PDP) technique, we can provide the integrity of data in Multi cloud storage. The design and implementation of an efficient PDP scheme for Multi cloud storage at sector level to support the scalability of service and distributed data, in which the multiple cloud service providers to store and manage the client's data. Also the security is used based on multi-prover zero-knowledge proof system to fulfill the soundness, completeness and zero-knowledge properties. PDP scheme based on response for verification of the same type of data and hash index hierarchy. The security is provided for the cloud system using the multi-prover zero-knowledge proof system which can also satisfy completeness, knowledge soundness, and zero-knowledge properties, by providing lightweight PDP scheme based on cryptographic hash function and symmetric key encryption. The numbers of updates and challenges are limited and fixed in advance. Users cannot perform block insertions anywhere.

Recently, the cloud storage service became prominent area for research in market by providing a comparably low-cost, scalable, position-

independent platform for clients' data. Both open architectures and interfaces are used for building cloud computing system; its ability to include many cloud services together for high interoperability.

2.2 Privacy preserving:

To make storage services accountable for data loss, some protocols that allow a third-party auditor to periodically verify the data stored by a service and assist in returning the data intact to the customer. Most importantly, our protocols are privacy-preserving, in that they never reveal the data contents to the auditor. solution to provide storage service accountability is through independent, third-party auditing and arbitration. The customer and service enter into an agreement or contract for storing data in which the service provides some type of payment for data loss or failing to return the data intact, e.g. free prints, refunds, or insurance. In such an agreement, the two parties have conflicting incentives. The service provider, whose goal is to make a profit and maintain a reputation, has an incentive to hide data loss. On the other hand, customers are terribly unreliable, e.g. casual home users. Customers can innocently (but incorrectly) or fraudulently claim loss to get paid. Thus, we involve an independent, third party to arbitrate and confirm whether stored and retrieved data is intact.
A straightforward solution for maintaining privacy during audits is for the customer to encrypt his contents using symmetric-key encryption and keep those keys intact and secret from uninvited parties. Then, the auditor can use existing provably secure, challenge-response schemes on the encrypted contents. This solution is unsatisfactory because an unsophisticated customer is increasingly likely over time either to lose the keys and be unable to recover the contents, or to leak the keys. In our protocols we shift the burden of keeping these secret keys to a storage service. Since services are already in the business of maintaining customers' data and privacy, the keys are safer with them. Keeping the data content private from the service is optional.
A customer can keep the keys and encrypted data with the same service, thereby revealing the contents to that service and allowing it to provide value-added features beyond storage like search. Otherwise, the customer can separate the keys and encrypted data onto non-colluding services to maintain complete privacy.

The auditor is responsible for auditing and extracting both the encrypted data and the secret keys. Our protocols, however, never reveal the secret key to the auditor. Although we present the protocols for handling the encrypted data for completeness, they are straightforward extensions of existing techniques. Our main contributions are (1) in motivating and specifying the problem of privacy-preserving audit and extraction of digital data (2) privacy-preserving protocols for auditing and extracting the encryption key. In this paper, we present these protocols and show how they provably ensure data integrity and data privacy from the auditor.

2.3 Homomorphic finger print:

Erasure coding can reduce the space and bandwidth overheads of redundancy in fault-tolerant data storage and delivery systems. But it introduces the fundamental difficulty of ensuring that all erasure-coded fragments correspond to the same block of data. Without such assurance, a different block may be reconstructed from dif-ferent subsets of fragments. This paper develops a technique for providing this assurance without the bandwidth and computational overheads associated with current approaches. The core idea is to distribute with each fragment what we call *homomorphic fin-gerprints*. These fingerprints preserve the structure of the erasure code and allow each fragment to be independently verified as cor-responding to a specific block. We demonstrate homomorphic fin-gerprinting functions that are secure, efficient, and compact.

Systems in which clients cannot be trusted to encode and dis-tribute data correctly use one of two approaches. In the first ap-proach, servers are provided the entire block of data, allowing them to agree on the contents and generate their own fragments. Savings are achieved for storage, but bandwidth overheads are no better than for replication. In the second approach, clients verify all *n* fragments when they perform a read to ensure that no other client could observe a different value. In this approach, each fragment is accompanied by a cross-checksum which con-sists of a hash of each of the *n* fragments. A reader verifies the cross-checksum by reconstructing a block from *m* fragments and then recomputing the other (*n m*) fragments and comparing their hash values to the corresponding entries in the cross-checksum, a significant computational overhead.

This paper develops a new approach, in which each fragment is accompanied by a set of fingerprints that allows each server to independently verify that its fragment was generated from the orig-inal value. The key insight is that the coding scheme imposes cer-tain algebraic constraints on the fragments, and that there exist ho-momorphic fingerprinting functions that preserve these constraints. Servers can verify the integrity of the erasure coding as evidenced by the fingerprints, agreeing upon a particular set of encoded frag-ments without ever needing to see them. Thus, the two common approaches described above could be used without the bandwidth or computation overheads, respectively.

2.4 Replication protocol for byzantile failure:

Our growing reliance on online services accessible on the Internet demands highly available systems that provide correct ervice without interruptions. Software bugs, perator mistakes, andmalicious attacks are a major cause of service interruptions and they can cause arbitrary behavior,that is, Byzantine faults. This article describes a new replication algorithm, BFT, that can be used to build highly available systems that tolerate Byzantine faults. BFT can be used in practice to implement real services: it performs well, it is safe in asynchronous environments such as the Internet, it incorporates mechanisms to defend against Byzantine-faulty clients, and it recovers replicas proactively. The recovery mechanism allows the algorithm to tolerate any number of faults over the lifetime of the system provided fewer than 1=3 of the replicas become faulty within a small window of vulnerability. BFT has been implemented as a generic program library with a simple interface. We used the library to implement the first Byzantine-fault-tolerant NFS file system,BFS. The BFT library and BFS perform well because the library incorporates several important optimizations, the most important of which is the use of symmetric cryptography to authenticate messages. The performance results show that BFS performs 2% faster to 24% slower than production mplementations of the NFS protocol that are not replicated. This supports our claim that the BFT library can be used to build practical systems that tolerate Byzantine faults.

3 PROPOSED WORK

3.1 Architecture:
The cloud architecture mainly consists of three entities those are User, Cloud Service Provider and Third Party Auditor Users who have a large amount of data to be stored in multiple clouds and have the permissions to access and manipulate stored data; Cloud Service Providers (CSPs) who work together to provide data storage services and have enough storages and computation resources; and Third Party Auditor (TPA) who is trusted to store verification parameters and offer public query services for these parameters.

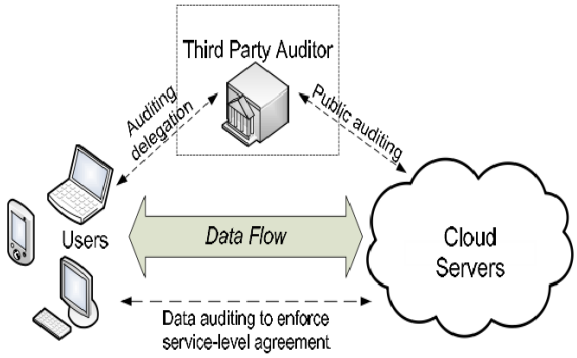

Figure: Cloud Storage Architecture

3.2 Design Goals:

To ensure the storage correctness in the cloud and security for our data which is stored in the cloud we need to design best matched designs for

secure dynamic data verification by third party auditor and we can reach the following goals: 1) Correctness of stored data:which will make sure the user to verify either his data is stored in the cloud is correct or not 2)Localizing the data error: Effectively locate the misbehaving server when any data corruption is occurred 3)Secure Dynamic data support: if user is doing any dynamic operations on the cloud then also we must need to provide the same level security as we provided to data in the cloud. 4) Lightweight communication: the users can perform their verification easily.

3.3 Implementation:

3.3.1 Securing stored and dynamic data in cloud:
As we know that the user can store his files in the cloud by buying a particular cloud service for some period of time according to their need. In our application we are providing security to the user's data which is stored in the cloud and also providing security when they are performing the dynamic operations on the stored text files of the user. The user can upload the data file into the cloud. That particular text file is divided in to three equal blocks which are named as First Block, Middle Block and Last Block. Here we are computing a token key for each of the block by using token computation algorithm. This token computation is done by depending on the Secret key of the user and the predefined Rijandael class and the token keys are sent to the user's mail. We can provide this mail service by using SMTP. The secret key of the user can be generated by random selecting of the key from specicified range and also sent to the user's mail after login. After token key generation of each block the user has an option either to Allow or Block the Secret key: The secret key will be selected randomly. Selection of key from specified range and it sent to the user's mail if they are online they can check the mail and enters secret key to go forward if the user is offline then they can get secret key by clicking on the link get secret key which the link retrives the secret key by random selection of key from specified range. User can upload the text file.After uploading the file the file can be converted into three equal sized blocks which are named as First Block, Middle Block, Last Block.Here we have an option to select either Allow or Block the user's data to the third party auditor where as selecting allow option allows the Third party auditor can read the original data of user if the user selects the option as Block then the data will be encrypted. The encryption process uses Rijandeal class. The token key will be generated to each block separately. The token key generation is also depends on random selection of a number from user specified range. When the user selects Block option the data and the token key both will be encrypted by using Rijandeal class which is worked as AES(Advanced Encryption Standard) but the difference is that AES has particular key sizes where as Rijandael has no particular size for key generartion.After the user has Finished his upload process the mail will be sent to user with all token keys. If the user needs to perform any further operation on the uploaded file then user can select file process. The file process is one which allows user to perform dynamic operations on cloud stored data. User can update,delete or append the data to the original file, the status of file process also sent to user's mail by using SMTP.
Third party auditor is also one of the server which are connected to the cloud, the purpose of selecting tpa is that if the user doesn't have time to verify the correctness of data in cloud then they can select the tpa and the duties of tpa is that verifying the storage correctness of user's data by comparing the source data and the stored data in multiple connected servers in the cloud if the data is same then we can say that as secure if not the particular server which is not given matched information that will be the misbehaved server. Even the third party also doesn't know the original source code why because the user can encrypt the data and also token keys. This encryption was done by using predefined Rijandael class as it is worked as AES algorithm.

4. RESULTS AND ANALYSIS:

In the cloud computing services the security is the major issue. To overcome some of the security issues like malicious data modification attack and server colluding attacks here we developed secure dynamic data support and trusted third party verification of user's file in the cloud. The secure dynamic data support will be done by generating the individual token keys for each block of data and sending that keys to your mail. This token key is generated by randomly selecting the number from specified range. The token key can be used to process the user file when performing operations like uploading. Appending and deleting. So that we are providing security to the dynamic data operations by sending token keys to user's mails.

text in the block as well as token key. If the user selects Block option then the text in the file is encrypted by using Rijandael algorithm. So that the even the third party auditor also can't see the user's original text to verify the storage correctness of user's file in the cloud. If the user feels burden to check and verify the storage correctness of the data then they can assign that work to the third party auditor so that the TPA can verify the user files.

3.3.2 Trusted Third Party Auditor(TPA):

The third party auditor is one who is having the capabilities if doing the cloud operations on user's request. Third party auditor is also a cloud server which is connected with remaining all distributed cloud servers so that TPA can able to verify the user files in the cloud either any malfunctioning or misbehaving server is corrupting the users data by proving the challenge which contains size of the file and token keys of the file to the distributed cloud servers. TPA stores the user's file related tokens and encrypted data and compares the received challenge results from different servers if both are same then there is no malfunctioning in any server if any one server gives not equal results then it we can easily identify that is the misbehaving server malfunctioning is occurred in that particular server. To verify the user file storage correctness also we are using an algorithm which is named as Storage Correctness Verifier. After verifying the file the third party auditor the status of the file can sent to the user's mail.We are providing a trusted auditing mechanism why means the TPA can only see the user file's encrypted text and encrypted token keys its enough to check or verify the storage correctness of the file.
Third party auditor is also one of the server which are connected to the cloud, the purpose of selecting tpa is that if the user doesn't have time to verify the correctness of data in cloud then they can select the tpa and the duties of tpa is that verifying the storage correctness of user's data by comparing the source data and the stored data in multiple connected servers in the cloud if the data is same then we can say that as secure if not the particular server which is not given matched information that will be the misbehaved server. Even the third party also doesn't know the original source code why because the user can encrypt the data and also token keys. This encryption was done by using predefined Rijandael class as it is worked as AES algorithm.

5. CONCLUSION:

In this paper, we investigate the problem of data security in cloud data storage, which is essentially a distributed storage system. To achieve the assurances of cloud data integrity and availability and enforce the quality of dependable cloud storage service for users, we propose an effective and flexible distributed scheme with explicit secure dynamic data support, including block update, delete, and append. We rely on the homomorphic token with distributed verification by trusted third party auditor data, our scheme achieves the of storage correctness insurance and data error localization, i.e., whenever data corruption has been detected during the storage correctness verification across the distributed servers, we can almost guarantee the simultaneous identification of the misbehaving servers. Considering the time, computation resources, and even the related online burden of users, we also provide the extension of the proposed main scheme to support multimedia data and integrity checking tasks and be worry-free to use the cloud storage services. Through detailed security and extensive experiment results, we show that our scheme is highly efficient and resilient to Byzantine failure, malicious data modification attack, and even server colluding attacks.

6. ACKNOWLEDGEMENT:

I O.Rajitha pursuing M.Tech in the Computer science department in MITS.Students work is incomplete until they thank the almighty & his teachers. I sincerely believe in this and would like to thank Dr.S.Murali Krishna,B.Tech,M.Tech,Ph.D,Professor & Head Of the Department of Computer Science, MITS, Madanapalle for his encouragement, motivation and for guiding me to write this paper.

REFERENCES:

[1] C. Wang, Q. Wang, K. Ren, and W. Lou, "Ensuring data storage security in cloud computing," in Proc. of IWQoS'09, July 2009.
[2] Amazon.com, "Amazon web services (aws)," Online at http://aws.amazon.com/, 2009.

[3] Q. Wang, C. Wang, J. Li, K. Ren, and W. Lou, "Enabling public verifiability and data dynamics for storage security in cloud computing" Springer-Verlag, Sep. 2009.

[4]G. Ateniese, R. D. Pietro, L. V. Mancini, and G. Tsudik, "Scalable and efficient provable data possession," in Proc. of SecureComm'08,2008.

[5] Sun Microsystems, Inc., "Building customer trust in cloud computing with transparent security," Online at https://www.sun.com/offers/details/sun ransparency.xml, November 2009.

[6] M. A. Shah, R. Swaminathan, and M. Baker, "Privacy-preserving audit and extraction of digital contents," Cryptology ePrint Archive, Report 2008.

[7] Q. Wang, K. Ren, W. Lou, and Y. Zhang, Dependable and secure sensor data storage with dynamic integrity assurance," in *roc. IEEE INFOCOM'09*, Rio de Janeiro, Brazil, Appril 2009.

[8] M. Castro and B. Liskov, "Practical byzantine fault tolerance and proactive recovery," *ACM Transaction on Computer Systems*, vol. 20, no. 4, pp. 398–461, 2002.

[9] C.Wang, K. Ren, W. Lou, and J. Li, "Towards publicly auditable secure cloud data storage services," *IEEE Network Magazine*, vol. 24, no. 4, pp. 19–24, 2010.

[10] C. Erway, A. Kupcu, C. Papamanthou, and R. Tamassia, "Dynamic provable data possession," in *Proc. of CCS'09*, 2009, pp. 213–222.

[11] 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/.