International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395 -0056
Volume: 02 Issue: 09 | Dec-2015 www.irjet.net p-ISSN: 2395-0072

PERFORMANCE OF CLOUD DATA INTEGRITY PROOFS IN CLOUD STORAGE SYSTEM USING CRYPTOGRAPHIC KEY

B.M.RAJESH[1]

Asst Prof, Department of Information Technology, SNMV College of Arts and Science, Tamil Nadu, India

---

Abstract - ***The de-facto solution to the increasing storage costs of IT Enterprises is the Cloud computing has been envisioned. In order to frequently update their hardware with the high costs of data storage devices as well as the rapid rate of which data is being generated it proves costly for enterprises or individual users. Data outsourcing to the cloud leads to reduction in storage costs also helps in reducing the maintenance. In this scheme a proof of data integrity in the cloud which the customer can employ to check the correctness of his data in the cloud is provided. Both the cloud and the customer can greed upon this proof and can be incorporated in the Service level agreement (SLA) .***

Data is moved to a remotely located cloud server in cloud computing. Whenever needed the Cloud faithfully stores the data and return back to the owner. The data stored in the cloud is secured and not altered by the cloud or Third Parity Auditor (TPA) is not guaranteed. Apart from reducing the storage costs cloud data outsourcing to the cloud also helps in reducing the maintenance. The user does not have any control on the Cloud storage which moves the user's data to large data centers, which are remotely located. At the client side the storage is minimal which will be useful for thin clients is ensured in this research.

Cloud stores the data faithfully and returns back to the owner whenever needed. The data stored in the cloud is no guarantee that it is secured and not altered by the cloud or Third Party Auditor (TPA). The user must be able to use the assist of a TPA in order to overcome the threat of integrity of data. The TPA has experience in checking integrity of the data, that clouds users does not have, and that is difficult for the owner to check. The content in the cloud should be correct, consistent, accessible and high quality. The aim of this paper is twofold 1) ensuring the integrity of the data and Provides the proof that data is in secured manner. 2) Providing Cryptographic key to secure the data in the cloud. The implementation and the test results are promising in the proposed approach.

1. INTRODUCTION

Data outsourcing to cloud storage servers is raising trend among many firms and users owing to its economic advantages. This essentially means that the owner (client) of the data moves its data to a third party cloud storage server which is supposed to - presumably for a fee - faithfully store the data with it and provide it back to the owner whenever required.

Though storing of user data in the cloud has its advantages it has many interesting security concerns which need to be widely investigated for making it a consistent solution to the problem of avoiding local storage of data. In this paper we have taken the problem of implementing a protocol for obtaining a proof of data possession in the cloud occasionally referred to as Proof of retrievability (POR).This problem tries to acquire and confirm a proof that the data that is stored by a user at a remote data storage in the cloud is not modified by the archive and thereby the integrity of the data is assured.

One primary feature of this pattern irregular is that data are being centralized and outsourced into clouds. The cloud storage service (CSS) reduces the burden of storage management and maintenance. However, if such an significant service is vulnerable to attacks, it would bring permanent losses to users since their data or archives are stored into an undecided storage collection outside the enterprises. The following reasons leads to security risks : When compared to personal computing devices the cloud infrastructures are much more powerful and reliable. For the benefits of their control they are still subject to security threats both from outside and inside the cloud, to behave falsely toward the cloud users there exist various motivations for cloud service providers (CSP) furthermore, the dispute occasionally suffers from the lack of trust on CSP.

The solutions to review the rightness of the data in a cloud surroundings can be difficult and costly for the cloud users. Therefore, it is fundamental to understand public audit ability for CSS, so that data owners may way out to a Third Party Auditor (TPA), who has knowledge and capabilities that a common user does not have, for periodically auditing the outsourced data. This audit service is considerably significant for digital forensics and data assurance in clouds.

To apply public audit ability, the notions of proof of retrievability (POR) and Provable Data Possession (PDP) have been planned by some researchers. Their method

was based on a probabilistic proof technique for a storage provider to prove that clients' data remain intact without downloading the stored data, which is called "verification without downloading".

In this paper we deal with the problem of implementing a protocol for obtaining a proof of data possession in the cloud sometimes referred to as Proof of retrievability (POR).This Problem tries to obtain and verify a proof that the data that is stored by a user at a remote data storage in the cloud (called cloud storage archives or simply archives) is not modified by the archive and thereby the integrity of the data is assured. Such kinds of proofs are very much helpful in peer-to-peer storage systems, network file systems, long-term archives, web-service object stores, and database systems. Such verification systems prevent the cloud storage archives from misrepresenting or modifying the data stored at it without the consent of the data owner by using frequent checks on the storage archives. Such checks must allow the data owner to efficiently, frequently, quickly and securely verify that the cloud archive is not cheating the owner.

Cloud Storage

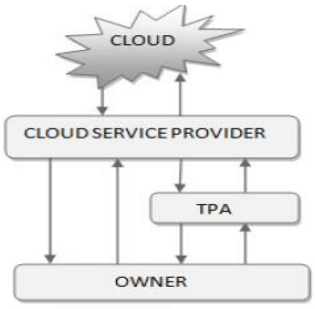


Fig 1 . Architecture of Cloud Storage

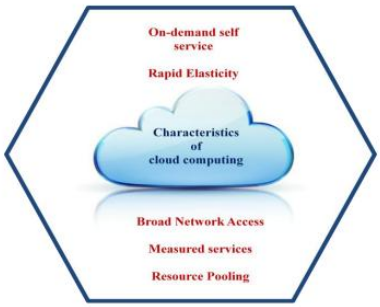


Fig 2. The Characteristics of Cloud Computing

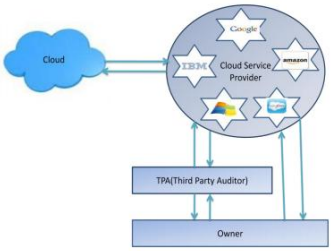


Fig 3. Cloud service providers

Cloud Computing can be defined as a computing paradigm that provides dynamic computing environment for end users that is reliable and customized and also guarantees quality of service (QoS). Being synonymous to "Internet" it provides service oriented applications in form of infrastructure as a service (IaaS), platform as a service (PaaS), and software as a service (SaaS) over internet. Cloud computing is independent of Grid computing and Utility computing but when cloud computing is implemented publically then it is done as Utility computing. Cloud provides the facility of renting the services instead of buying or purchasing them and stores the complete data at its data centers for future access. The basic components of cloud computing are Clients, Services, Application, Platform, Storage and infrastructure. The essential characteristics a cloud must possess are: On-demand self service¬ Broad Network Access¬ Rapid Elasticity¬ Resource Pooling¬ Measured services¬ Cloud computing with its acceptance also has some growing needs which affect the complete working of cloud, and one of those needs is the need for "security". Cloud at present is lacking in its security needs in terms of data integrity, authorization and confidentiality. The clouds at present are being provided by specific vendors like Amazon and Google .

2. RELATED WORK

Ari Juels and Burton S. Kaliski Jr proposed a scheme called Proof of retrievability for large files using "sentinels"[3]. In this scheme, unlike in the key-hash approach scheme, only asingle key can be used irrespective of the size of the file or the number of files whose retrievability it wants to verify. Also the archive needs to access only a small portion of the file F unlike in the key-has scheme which required the archive to process the entire file F for each protocol verification. This small portion of the file F is in fact independent of the length of F. The schematic view of this approach In this scheme special blocks (called sentinels) are hidden among other blocks in the data file F. In the setup phase, the verifier randomly embeds these sentinels among the data blocks. During the verification phase, to check the integrity of the data file F, the verifier challenges the prover (cloud archive) by specifying the positions of a collection of sentinels and asking the prover to return the associated sentinel values.

The simplest Proof of derivability (POR) scheme can be made using a keyed hash function hk(F). In this scheme the verifier, before archiving the data file F in the cloud storage, pre-computes the cryptographic hash of F using hk(F) and stores this hash as well as the secret key K. To check if the integrity of the file F is lost the verifier releases the secret key K to the cloud archive and asks it to compute and return the value of hk(F). By storing multiple hash values for different keys the verifier can check for the

integrity of the file F for multiple times, each one being an independent proof. Though this scheme is very simple and easily implementable the main drawback of this scheme are the high resource costs it requires for the implementation. At the verifier side this involves storing as many keys as the number of checks it want to perform as well as the hash value of the data file F with each hash key. Also, computing hash value for even a moderately large data files can be computationally burdensome for some clients. As the archive side, each invocation of the protocol requires the archive to process the entire file F. Also the archive needs to access only a small portion of the file F unlike in the key-has scheme which required the archive to process the entire file F for each protocol verification. This small portion of the file F is in fact independent of the length of F. The schematic view of this approach is shown in Fig. 4. In this scheme special blocks are hidden among other blocks in the data file F. file in order to prevent from the unauthorized access.

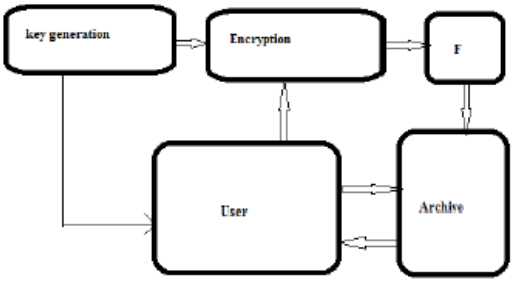


Fig 4. Schematic view of a POR

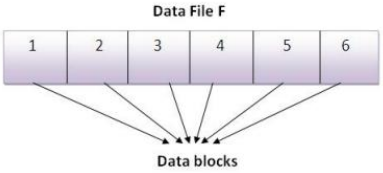
Juels and Kaliski[1] proposed a model Proofs of Retrievability(POR) was one of the first most important attempts to formulize the notion "guaranteed remotely and reliable integrity of the data without the retrieving of data file." It is basically a data encryption mechanism which detects data corruptions and retrieve the complete the data without any damage. Shacham and Waters[2] gave a new model for POR enabling verifiability of unlimited number of queries by user with reduced overhead. Later Bowels and Juels[3] gave a theoretical model for the implementation of POR, but all these mechanisms proposed were weak from the security point because they all work for single server. Therefore Bowels [4] in their further work gave a HAIL protocol extending the POR mechanism for multiple servers. Priya Metri and Geeta Sarote[5] proposed a threat model to overcome the threat of integrity and provide data privacy in the cloud storage. It uses TPA(Third Party Auditor) and digital signature mechanism for the purpose of reliable data retrievable. The TPA being used notifies any unauthorized access attempting to make changes, avoiding the changes in data and maintaining the originality of data. Atienies and Burns[6] gave Provable Data Possession(PDP) mechanism which verifies the integrity of data being outsourced, detecting all kind of errors occurring in data but doesn‟t guarantee complete data retrievable. In their later work Atienies and Pietro[7] proposed a scheme which overcome all problems in PDP, but the main and basic problem on both proposed system didn‟t overcome was they work on single server. Therefore, later Curtmola[8] proposed a scheme to ensure data reliability and retrievability of data for multiple servers. Many mechanisms has been proposed till now to guarantee and ensure complete data integrity and data privacy of cloud storages based on encryption and cryptographic mechanisms using hash values and data encoding. Filho[9] proposed a RSA-based hash data integrity mechanism for peer-to-peer file sharing networks and exponentiation of complete data file is done, but this mechanism can be followed for the files and data of large size, and also this mechanism focuses on the static data files and not on files being dynamic in nature having localization problem. Atan and Abdullah[10] proposed a data integrity mechanism of Cloud Zone which uses the concept of Multi Agent System(MAS) architecture.

Juels and Kaliski[1] proposed a prototypical Proofs of Retrievability(POR) was one of the first most significant efforts to formulize the notion "guaranteed remotely and dependable integrity of the data lacking the retrieving of data file." Therefore Bowels [4] in their further work gave a HAIL protocol extending the POR mechanism for multiple servers. Priya Metri and Geeta Sarote[5] proposed a risk model to overcome the threat of integrity and provide data privacy in the cloud storage. It uses TPA(Third Party Auditor) and digital signature mechanism for the purpose of reliable data retrievable. The TPA being used notifies any illegal access attempting to make changes, avoiding the changes in data and maintaining the originality of data. Atienies and Burns[6] gave Provable Data Possession(PDP) mechanism which verifies the integrity of data being outsourced, noticing all kind of errors occurring in data but doesn‟t guarantee complete data retrievable. In their later work Atienies and Pietro[7] planned a scheme which overcome all problems in PDP, but the main and basic problem on both proposed system didn‟t overwhelmed was they work on single server. Filho[9] proposed a RSA-based hash data integrity mechanism for peer-to-peer file sharing networks and exponentiation of complete data file is done, but this mechanism can be followed for the files and data of large size, and also this mechanism focuses on the static data files and not on files being dynamic in nature having localization problem.

Atan and Abdullah [7] proposed a data integrity mechanism of cloud zone which uses the concept of Multi Agent System (MAS) architecture [4]. MAS are the techniques used in artificial intelligence where they communicate with each other to find a complete solution [4]. All the mechanisms proposed till now and discussed above —provide the services in cloud storages (CDSs) for ensuring the integrity of data transmission.

3. METHODOLOGY

We present a scheme which does not involve the encryption of the whole data. We encrypt only few bits of data per data block thus reducing the computational overhead on the clients.



In our data integrity protocol the verifier needs to store only a single cryptographic key - irrespective of the size of the data file F- and two functions which generate a random sequence. The verifier does not store any data with it. The verifier before storing the file at the archive, preprocesses the file and appends some meta data to the file and stores at the archive. At the time of verification the verifier uses this meta data to verify the integrity of the data. In order to prevent such modifications or deletions other schemes like redundant storing etc, can be implemented which is not a scope of discussion in this paper.

3.1 A DATA INTEGRITY PROOF IN CLOUD BASED ON SELECTING RANDOM BITS IN DATA BLOCKS

The client before storing its data file F at the client should process it and create suitable Meta data which is used in the later stage of verification the data integrity at the cloud storage. When checking for data integrity the client queries the cloud storage for suitable replies based on which it concludes the integrity of its data stored in the client. A. Setup phase Let the verifier V wishes to the store the file F with the archive. Let this file F consist of n file blocks. We initially preprocess the file and create metadata to be appended to the file. Let each of the n data blocks have m bits in them. A typical data file F which the client wishes to store in the cloud is shown in. The initial setup phase can be described in the following steps:

1)Generation of meta-data: Let g be a function defined as follows,

Where k is the number of bits per data block which we wish to read as Meta data. The function g generates for each data block a set of k bit positions within the m bits that are in the data block. Hence g(i, j) gives the j th bit in the i th data block. The value of k is in the choice of the verifier and is a secret known only to him. Therefore for each data block we get a set of k bits and in total for all the n blocks we get $n * k$ bits. Let mi represent the k bits of Meta data for the i th block. Figure 3 shows a data block of the file F with random bits selected using the function g.

2) Encrypting the meta data: Each of the meta data from the data blocks mi is encrypted by using a suitable algorithm to give a new modified meta data Mi . Without loss of generality we show this process by using a simple XOR operation. Let h be a function which generates a k bit integer αi for each i. This function is a secret and is known only to the verifier V,For the meta data (mi) of each data block the number αi is added to get a new k bit number Mi .

In this way we get a set of n new meta data bit blocks. The encryption method can be improvised to provide still stronger protection for verifiers data.

3) Appending of meta data: All the meta data bit blocks that are generated using the above procedure are to be concatenated together. This concatenated meta data should be appended to the file F before storing it at the cloud server. The file F along with the appended meta data Fe is archived with the cloud. Figure 4 shows the encrypted file Fe after appending the meta data to the data file F.

Verification phase let the verifier V wants to verify the integrity of the file F. It throws a challenge to the archive and asks it to respond. This Meta data will be a k-bit number. Hence the cloud storage server is required to send k+1 bits for verification by the client. The Meta data sent by the cloud is decrypted by using the number αi and the corresponding bit in this decrypted Meta data is compared with the bit that is sent by the cloud. Any mismatch between the two would mean a loss of the integrity of the client's data at the cloud storage.

By keeping project goals in mind, here we put the new scheme which guarantees the security of cloud data. The protocol developed chains public auditability with dynamic data operations. The planned system enables public auditability without recovering block of data from file for this we uses Homomorphism authentication method that was used in preceding model. There is the unforgivable metadata generator computed from separate data blocks. In the upcoming work two authenticators such as BLS signature [3] based authenticator. The safety mechanism is further described here. The procedure of protocol is divided into 1) Public auditability for storage correctness declaration: To allow anyone, not just the clients who originally deposited the file on cloud servers, to have the capability to verify the accuracy of the stored data on demand 2) Dynamic data operation support: This will allow the clients to do block-level operations on the data files while preserving the same level of data correctness assurance. The design should be as efficient as conceivable so as to ensure the seamless integration of public audit ability and dynamic data action support.

1) Setup In this stage KeyGen() method is raised to generate public key and private key. SigGen() is meant for pre-processing and Homomorphic authenticators and along with meta data. The SigGen() method takes two

arguments namely secret key and file. The file gratified is divided into blocks. Then signature is computed for each block. Each block's hash code is taken and two nodes' hash is merged into one in order to produce the next node. This process continues for all leaf nodes until tree node is originated. The root element is then taken by client and signs it and send to cloud storage server.

2) Default integration verification the content of outsourced data can be verified by either client or TPA. This is done by stimulating server by giving some file and block randomly. Up on the test, the cloud storage server computes the root hash code for the assumed file and blocks and then returns the computed root hash code and first stored hash code along with signature. Then the TPA or customer uses public key and private key in order to decrypt the gratified and compare the root hash code with the root hash code returned by clients.

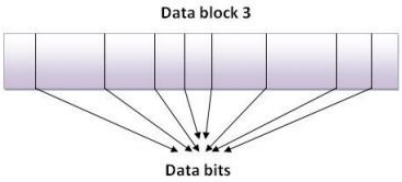


Fig 5. A data file F with 6 data blocks

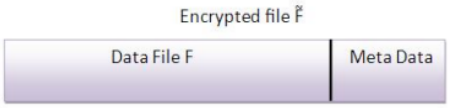
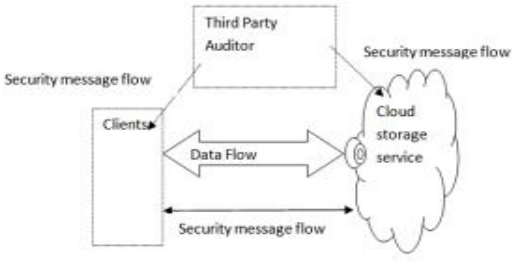


Fig 6. The encrypted file Fe which will be stored in the cloud.



The content of outsourced data can be confirmed by either client or TPA. This is done by challenging server by giving some file and chunk randomly. Up on the test, the cloud storage server computes the root hash code for the given file and blocks and then returns the calculated root hash code and originally deposited hash code along with signature. Then the TPA or client uses public key and private key in order to decrypt the content and match the root hash code with the root hash code returned by clients. 3.1 Algorithms Working Algorithm for data integrity verification Start/begin

- TPA creates random set.
- CSS calculates root hash code based on the file
- Name input. CSS calculates the originally stored value.
- TPA decrypts the given content and matches with• generated root hash. After verification, the TPA can regulate whether
- The integrity is breached. Stop

.• Data modifications are the recurrent operations on cloud storage. It is a process of substituting specified blocks with new ones. The data alteration operation can't affect the reason structure of client's data. Another process is known as data insertion. Data Insertion is a process of inserting new record in to existing data. The new blocks are inserted into specified locations or blocks in the data file F.

- Starts/begin
- Clients produce new Hash for tree then sends to CSS.
- CSS updates F and calculates new R'
- Client computes R, Clients confirms signature. If it fails output is FALSE. Compute new R and verify the update• If prove succeed, CSS update R'.
- Stop

Multi Agent System (MAS) Architecture for CDS Integrity: MAS architecture is a mechanism being developed from the concept of Multi Agent system (MAS) in artificial intelligence defined as "loosely-coupled network of entities that work together to find solutions for the problems which are beyond the knowledge of single entity". It is implemented on basic cloud architecture and consists of two main layers as cloud resource layer (cloud server side) and MAS architecture layer (cloud client-side). MAS Architecture layer known as cloud zone consists of 5 agents but most widely used agents for integrity are Cloud Service Provider Agent (CSPA) and Cloud Data Integrity Backup Agent (CDIBA).

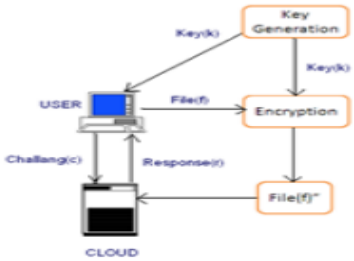


Fig 7. Cloud system

3.2 PROPOSED METHOD

The block which we are seen in below the encryption block this block represent metadata generated file. We can create metadata for file by selecting fewer bytes from it. In our system we can encrypt only fewer bits of file we cannot encrypt whole file so for selecting fewer bits we can create metadata. This metadata is useful for checking integrity of file which is aim of our system. According to the bit position we can check integrity of file. If bit position is varies form before uploading file and after Uploading or accessing of file then we can say that our file is modified or deleted. Now the next component is cloud, cloud is storage area which we store our file. When we store file communication link is establish between the users and cloud. When user finally store their file to the cloud then it send request to the cloud for conforming integrity of file in architecture we mention challenge(c) it means users challenges to the cloud for maintaining integrity with respect to that file. Cloud then send response to the users in the form of acknowledgement which says our file is securely stored to the cloud.

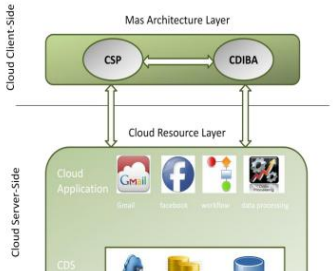


Fig 8. MAS Architecture "CLOUD ZONE"

The MAS architecture uses Prometheus methodology for designing the prototype and Java Agent Development Framework Security (JADE-S) for its implementation. Data Encoding Mechanism Data encoding is one of the basic methodologies used for the purpose of integration of data being transmitted.

We in our methodology are proposing a mechanism where these two mechanisms are combined, that is, MAS architecture for CDS and data encoding using hash values are combined together to give a new mechanism. MAS architecture basically uses two agents in client side layer for data integrity, CSPA which takes care of all the backing up of data in cloud zone and generate regular alarms if any error occurs, but doesn't guarantee any authorization as it only talks about the data which is being entered into cloud storage ad not about the data accessed by the clients regularly, that is, if there is any unauthorized access to data and data is being entered in correct format, then the cloud will provide the data. So to ensure that the data being entered is integrated and data being accessed from cloud is also integrated, we are combining these two mechanisms. This can be done inserting a hash value concept in CDIBA agent of MAS architecture.

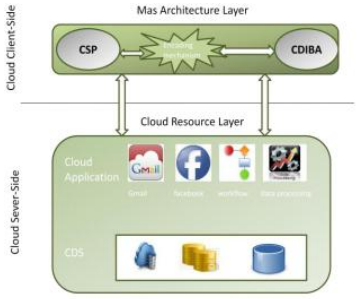


Fig 9. Encoded MAS Architecture

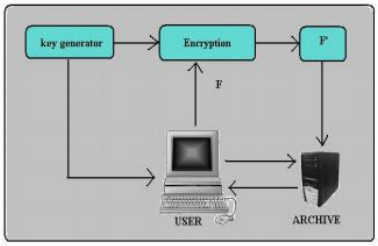


Fig 10. Semantic view of POR

The main role of sentinels is cloud needs to access only a small portion of the file (F) instead of accessing entire file. Sravan and saxena[6] proposed a Schematic view of a proof of retrievability based on inserting random sentinels in the data file.

We have worked to facilitate the client in getting a proof of integrity of the data which he wishes to store in the cloud storage servers with bare minimum costs and efforts. Our scheme was developed to reduce the computational and storage overhead of the client as well as to minimize the computational overhead of the cloud storage server. We also minimized the size of the proof of data integrity so as to reduce the network bandwidth consumption. At the client we only store two functions, the bit generator function g, and the function h which is used for encrypting the data. Hence the storage at the client is very much minimal compared to all other schemes [4] that were developed. Hence this scheme proves advantageous to thin clients like PDAs and mobile phones. The operation of encryption of data generally consumes a large computational power.

In this paper we provide a scheme which gives a proof of data integrity in the cloud which the customer can employ to check the correctness of his data in the cloud. This proof can be agreed upon by both the cloud and the customer and can be incorporated in the Service level agreement (SLA).

In this module we provide an efficient and secure cryptographic interactive audit scheme for public audit ability. We provide an efficient and secure cryptographic interactive retains the soundness property and zero-knowledge property of proof systems.

3.2.1 Data Storage Service System

In this module, we considered FOUR entities to store the data in secure manner:

1.Data owner (DO): Who has a large amount of data to be stored in the cloud.

2.Cloud service provider (CSP) : Who provides data storage service and has enough storage spaces and computation resources.

3.Third party auditor (TPA): Who has capabilities to manage or monitor – outsourced data under the delegation of data owner?

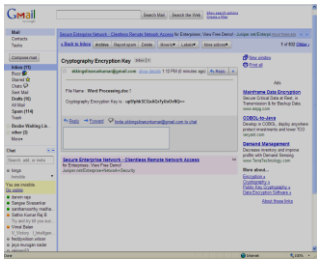
4. Granted applications (GA): Who have the right to access and manipulate stored data. These applications can be either inside clouds or outside clouds according to the specific requirements.

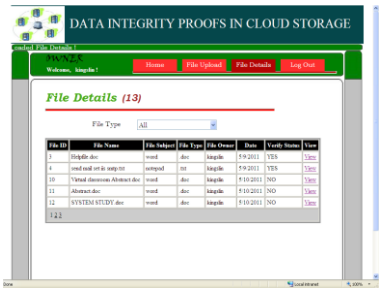
3.2.2 Audit Outsourcing Service System

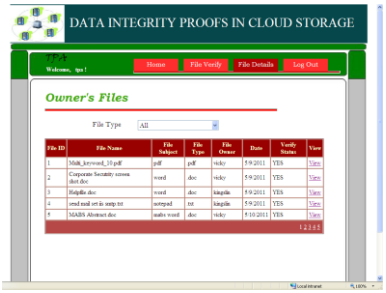
In this module the client (data owner) uses the secret key to preprocess the file, which consists of a collection of blocks, generates a set of public verification information that is stored in TPA, transmits the file and some verification tags to Cloud service provider CSP, and may delete its local copy.At a later time, using a protocol of proof of retrievability, TPA (as an audit agent of clients) issues a challenge to audit (or check) the integrity and availability of the outsourced data in terms of the public verification information. It is necessary to give an alarm for abnormal events.

ADVANTAGES OF PROPOSED SYSTEM:

- Apart from reduction in storage costs data outsourcing to the cloud also helps in reducing the maintenance.
- Avoiding local storage of data.
- By reducing the costs of storage, maintenance and personnel.
- It reduces the chance of losing data by hardware failures.
- Not cheating the owner.







4. CONCLUSION

In this paper we have worked to provide facility to the client which gets proof of integrity of the data which he wishes to store in the cloud storage. By doing this client can get advantage with respect to the cost and efforts. The Purpose of our scheme is to reduce computational and storage overhead at the user side and as well as reduce overhead of the cloud storage server. Our scheme is also reduce network bandwidth consumption. At the user side we use two functions first is bit generator and second is function which is used for encrypting the data. So therefore as compare to all other scheme our system provide minimal storage at the user side. Our scheme is suitable for small memory device like mobile phone. Large computational power is reduced because of encryption of file. We encrypt only few bits of file so computational time is saving of the client.

In this paper we discussed about cloud computing and the role of cloud storages (CDSs) in cloud computing, and describing the most important security threat of CDS which is data integrity and data privacy, the proposed mechanisms for integrity assurance and the problems being faced in these mechanism. Later we discussed about MAS architecture and data encoding using hash values for the purpose of integrating data transmission as well data integrity of CDS by encrypting and hashing the data in CDS using CDIBA agent of Cloud Zone.

REFERENCES

[1] E. Mykletun, M. Narasimha, and G. Tsudik, “Authentication and integrity in outsourced databases,” Trans. Storage, vol. 2, no. 2, pp. 107–138, 2006.

[2] D. X. Song, D. Wagner, and A. Perrig, “Practical techniques for searches on encrypted data,” in SP ’00: Proceedings of the 2000 IEEE Symposium on Security and Privacy. Washington, DC, USA: IEEE Computer Society, 2000, p. 44.

[3] A. Juels and B. S. Kaliski, Jr., “Pors: proofs of retrievability for large files,” in CCS ’07: Proceedings of the 14th ACM conference on Computer and communications security. New York, NY, USA: ACM, 2007, pp.584–597.

[4] 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.

[5] R. Sravan kumar and Saxena ,”Data integrity proofs in cloud storage” in IEEE 2011.

[6] E. Mykletun, M. Narasimha, and G. Tsudik, “Authentication and integrity in outsourced databases,” Trans. Storage, vol. 2, no. 2, pp. 107–138, 2006.

[7] D. X. Song, D. Wagner, and A. Perrig, “Practical techniques for searches on encrypted data,” in SP ’00: Proceedings of the 2000 IEEE Symposium on Security and Privacy. Washington, DC, USA: IEEE Computer Society, 2000, p. 44.

[8] A. Juels and B. S. Kaliski, Jr., “Pors: proofs of retrievability for large files,” in CCS ’07: Proceedings of the 14th ACM conference on Computer and communications security. New York, NY, USA: ACM, 2007, pp.584–597.

[9] A. Juels and B.S. Kaliski, Jr., “Pors: proofs of retrievability for large files,” in CCS‟07: Proceedings of the 14th ACM conference on Computer and communications security.

[10] H. Shacham and B. Waters, “Compact Proofs of Retrievability,” In Proceedings.of Asiacrypt ‟08, Dec. 2008.

[11] K. D. Bowers, A. Juels, and A. Oprea, “Proofs of Retrievability: Theory and Implementation,” Cryptology ePrint Archive, Report 2008/175,2008

[12] K. D. Bowers, A. Juels, and A. Oprea, "HAIL: A High-Availability and Integrity Layer for Cloud Storage," Cryptology ePrint Archive, Report 2008/489, 2008.
[13] Jia Xu and Ee-Chien Chang, "Towards efficient proofs of irretrievability in cloud storage".
[14] [14] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson,and D. Song, "Provable Data Possession at Untrusted Stores,"In Proceedings. Of CCS "07, pp. 598–609, 2007.
[15] G. Ateniese, R. D. Pietro, L. V. Mancini, and G. Tsudik, "Scalable and Efficient Provable Data Possession," In Proceedings of SecureComm "08, pp. 1–10, 2008.
[16] R. Curtmola, O. Khan, R. Burns, and G. Ateniese, "MR-PDP: Multiple-Replica Provable Data Possession," In Proceedings of ICDCS "08, pp.411–420,2008.
[17] D.L. Gazzoni Filho, and P. Barreto, "Demonstrating Data Possession and Uncheatable Data Transfer," Book Demonstrating data possession and uncheatable data transfer, Series Demonstrating data possession and uncheatable data transfer, ed., Editor ed.^eds., Citeseer, 2006, pp. 150.

AUTHOR

Mr.B.M.Rajesh has completed his M.Phil and having teaching experience of 9 years in SNMV College of Arts and Science. His area of specialization is Network Security and cryptography.