ISSN:1991-8178

Australian Journal of Basic and Applied Sciences

Journal home page: www.ajbasweb.com

Data Integrity Verification in Cloud Storage by Employing Public Verifiability

[1]T. Subha and [2]Dr. S. Jayashri

[1]*Research Scholar, Department of Information and Communication Engineering, Anna University, Chennai, India, Box:600025.*
[2]*Director- Research, Adhiparasakthi Engineering College, Melmaruvathur, India*.

ARTICLE INFO

Article history:
Received 12 March 2015
Accepted 28 April 2015
Available online 5 May 2015

Keywords:
Audit; Clients; Cloud; Data; Meta data; Token

ABSTRACT

The industries and individual clients prefer to store their data in the cloud storage server since they are relieved from the burden of storage maintenance, investment cost. When the data is outsourced, ensuring the integrity of the data stored at cloud server is of paramount importance. We propose a method to check the integrity and availability of the data stored in cloud server. This paper describes two functions used to create Meta data for the file blocks. This Meta data is served as a token to check the data integrity. On behalf of client the auditor (TPA – Third Party Auditor) performs the audit and check whether the data is altered by any means. The report containing the audit results is generated to the clients. This scheme achieves high efficiency and low communication cost.

© 2015 AENSI Publisher All rights reserved.

To Cite This Article: T. Subha and Dr. S. Jayashri., Data Integrity Verification in Cloud Storage by Employing Public Verifiability. ***Aust. J. Basic & Appl. Sci.,*** *9(11): 376-380, 2015*

INTRODUCTION

Cloud computing is next generation computing gaining more attention in the industrial, research and academic groups. This enables the IT industries to concentrate more on business. Juels and Kaliski (2007) quoted they are relieved from the burden of storage, maintenance and operational cost. These are delegated to the cloud service provider. Wang *et al*. (2009) stated that it offers services across the internet and some of the benefits are on demand resource access, scalability, can pay only for what we use, ability to scale up and down resources according to our need, elasticity, multi-tenancy etc.

Cloud computing provides three types of services (IaaS, SaaS, PaaS). All the services are provided immediately and instantly. Cloud provides three types of deployments models. They are public, private and hybrid model. Private model is solely owned and maintained by third party or by the particular organization. Public cloud is offered by the public providers like Amazon, Rackspace etc. It is accessible and operated by the public. Hybrid is the one which comprises the features of public and private cloud functions. The users who outsource their data to the cloud server fully depend on them for their data availability all the time. Even the best storage offered by Amazon's S3 had faced downtime was mentioned by ren *et al*.(2012) and Amazon.com.

Many scenarios had been addressed previously by Wang *et al*, where the users need to concern about their integrity and privacy of their data out sourced into cloud storage server. Arrington (2006) insisted on the importance of addressing the security key challenges while the data is outsourced to the third party server. It arises mainly because the user would not have control over their data; the location of the data where it is actually stored is not known. The cloud server themselves is not trustworthy, it may try to delete some files which are rarely accessed due to lack in storage space. In some cases failures in storage space is not informed to the clients to keep up the reputation of the cloud server. Wang *et al*. (2012) stated that there exists Legal and multi tenant issues.

Therefore it is necessary to design suitable cryptographic algorithms to verify the integrity and confidentiality of the outsourced data was given by Ateniese *et al*. (2008). Traditional algorithms need to be modified so as to satisfy the above requirements. Simply downloading the entire file for the sake of verifying the data integrity is time consuming and incurs high communication cost. Various encryption algorithms had been discussed and presented in previous works. The data could be encrypted and sent to the cloud storage server to ensure the confidentiality of the file. So no one can be able to view the data other than the clients. Authentication mechanisms can be included in order to assign

Corresponding Author: T. Subha, Research Scholar, Department of Information and Communication Engineering, Anna University, Chennai, India, Box:600126
E-mail: subharajan@gmail.com

credentials to the users in the groups to access and modify the content of the file.

Motivation:

Nowadays cloud storage service receives enormous attention towards outsourcing data. It provides less investment on cost, scalable and location independent platform for storing and managing client's data. Verifying the Data integrity, availability and confidentiality becomes the crucial security requirement in this case. Many mechanisms had been discussed, published in earlier papers and details are discussed in Related Works. Among the one is employing public verifiability to delegate the auditing task to the auditor. The auditor can be anyone who possesses the auditing capabilities for carrying out the audit on behalf of the clients / users. The auditor challenges the cloud server by sending random blocks of data in a file. He then verifies the response given by server and checks whether the integrity is maintained or not. Being motivated by these factors this paper aims to propose a methodology to provide low communication and computation cost data integrity verification scheme by employing public verifiability.

2. Related Work:

Many works have been previously published on verifying the integrity, confidentiality and availability of the data. The first one was to consider the work by Ateniese *et al*. (2007) proposed PDP (Provable Data Possession) public verifiability scheme, which allows anyone (authorized person) other than the user to challenge the storage server for data possession. They considered only static files. The PDP scheme found to be not secured against replay attacks in changing scenarios. To support dynamic data's the dynamic version of their protocol called Scalable PDP was proposed by Ateniese *et al*.(2008). They applied cryptographic hash functions and symmetric key encryption. The number of challenges and updates that can be performed on a query is limited and fixed in advance. Users are not allowed to perform insertions anywhere.

Erway *et al*. (2009) introduced the next scheme based on the scalable PDP called DPDP (Dynamic Provable Data Possession. It uses "blockless" scheme, but there is a possibility of data leakage by the response of a challenge. This is the main drawback of Erway Scheme. POR scheme (Proof of Retrievability) was presented by Juels and Kaliski (2007) for ensuring the files are retrievable and the possession of data. But it needs to pre-process the file first before sending the file to a CSP. This scheme fails to achieve for updating data. Wang *et al*. (2009) proposed the first dynamic scheme privacy preserving public auditing scheme that considers achieving public auditability, storage correctness. They were the one considered dynamic data's (insert, delete and update). They introduced batch auditing also. Their scheme is called privacy preserving since the user content is not disclosed to the auditor. The drawback of their scheme is its inability to resist against forgery attacks.

Sravan Kumar & Ashutosh Saxena (2011) proposed data integrity proofs in cloud storage. They provide a scheme that aggregates the proof by both the cloud and customer. It can be incorporated into Service Level Agreement (SLA). The drawback of their scheme is it applies only to static storage of data. It would not handle the cases where data is dynamically changing.

2.1 Our Contributions:

In our paper we propose to have two cryptographic hash functions to generate metadata. The generated metadata is again substituted with another simple function to encrypt the Meta data. The newly encrypted metadata provides stronger security since the bits are again rearranged. The secret is known only to the user. So it preserves privacy property also.

Organization:

We organize the paper as follows. In section 3, we state the system model of the proposed system, in section 4, we explain the integrity proof and section 5, deals dynamic data operations, and then we discuss implementation in section 6. Finally in section 7, we conclude with conclusion.

3. Proposed System:

The system model of our proposed system is shown in Fig.1. This system consists of three main entities.

Clients:

It refers to the users or individual clients, industries who wants to outsource their data.

Cloud Storage Server (CSS):

Cloud Storage Server is provided by the CSP (Cloud Service Provider) who has enormous amount of storage.

Third Party Auditor (TPA):

Authorized personnel has the capability to audit the data integrity on behalf of the client.

4. Integrity Proof:

We propose a method to check the integrity of the data stored at cloud storage server. The client needs to pre-process the file before it stores the data at the cloud server. It generates the Meta data based on the individual file. This data can be used at verification stage for checking the integrity of the data. Meta data is secret and known only to the client/users. Sometimes the storage server may alter the data blocks in client's data or may delete portions of data which are not frequently accessed by clients. This is done by the server due to minimum storage

space. In order to ensure the correctness and integrity of data the client performs data verification very often. Our algorithm achieves correctness and verification process. The algorithm is divided into four phases.

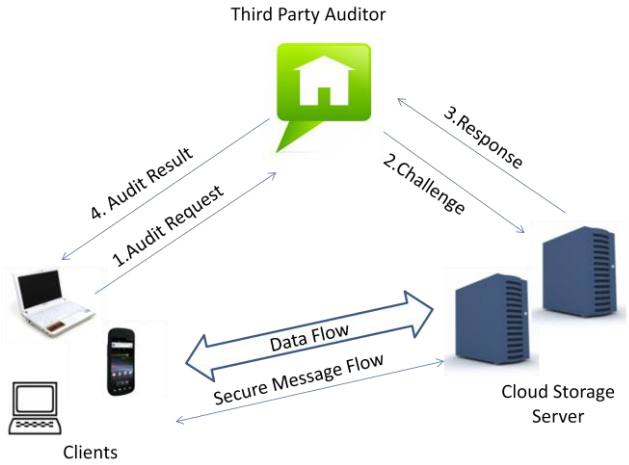

Fig. 1: System Model.

4.1 Setup phase:

Pre-processing of file F is carried out in this phase. Initially the data file F is divided into n data blocks. Each file block consists of n amount of bits. We apply the cryptographic hash function to generate the Meta data from the file blocks. Meta data is stored at the client side and used at later stage of data verification.

$F = \{M_1, M_2, \ldots\ldots\ldots\ldots\ldots, M_n\}$ (1)

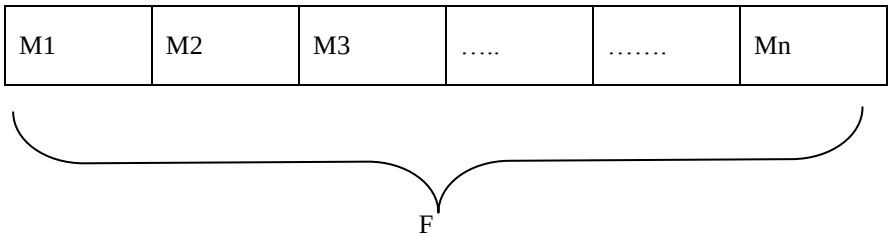

Fig. 2: File divided into n blocks

4.2 Tag generation phase:

Generation of Meta data known as tags can be generated in this phase. It uses a function to generate a smaller amount of t bits (which we assign as Meta data) from each data block. The t bits are called Meta data. We should try to have minimum number of bits as Meta data. Then only it will be stored using less space. We apply a function

$F: M \times T \longrightarrow N$ (2)

Here M is the no of data blocks of M, T is the amount of bits generated as tags and N is the amount of bits (n) in a particular data block of N. We define this function as

$f: \{0,1\}^m \times \{0,1\}^t \longrightarrow \{0,1\}^n$ (3)

Here M ε {m1, m2 …mn} -- no of data blocks in a file and

T ε {1 … t} – the amount of bits generated from the particular data block m_i

It takes input as the divided data blocks, the no of bits in each data block and produces the secret bits (t_i) as the tag. The no of bits generated as Meta data is known only to the user and is kept secret. The total no of bits generated as a metadata from n file blocks is $m_i * t$ where t is the amount of bits generated as Meta data & m_i is the no of data blocks.

In order to provide more security to the Meta data, we apply a simple encryption function to keep the Meta data more secure. Homomorphic PRF can be applied in many practical applications. According to Dan boneh *et al*. (2012) F is key homomorphic if given F (k1, x) and F (K2, x), it produces output F (k1 $\oplus$ K2, x). That is, a PRF is said to be homomorphic based on its key size. We apply homomorphic PRF to the tags generated in the previous step to produce new Meta data.

Here we apply partial Key homomorphic function. The function can be written as Pseudo Random Function. F (k1 $\oplus t_i$, R). Here R is a random value generated randomly using random oracle model. The key value is XOR ed with the Meta data ($t_{i,\ i=1.2.....n}$) value and the output is again concatenated with the random value generated to produce the new

Meta data (**T'**). The newly constructed Meta data is concatenated together with the original file F. If a user wants to keep the file secure and confident, the entire file with the Meta data can be encrypted using encryption algorithm. And then the file F is sent to the Cloud. The metadata is sent to the third party auditor which is later used for verification.

4.3 Challenge phase:

To make sure the integrity of the data the TPA sends a challenge to the storage server. The verifier specifies the position of the block, bit number t generated by the function, the position of the metadata corresponding to the file block. Then the prover sends back the response to the verifier.

4.4 Verification phase:

The verifier decrypts the Meta data sent by the storage server. It runs the algorithm to retrieve the original t bits. Then the data from the cloud server is compared against decrypted Meta data. If the result FAILS, the integrity of the data is lost. Otherwise ACCEPT. The integrity of the data is maintained by third party server. The auditing result is sent to the clients.

5. Dynamic Data Operations:

From user's point of view it is necessary to modify the data or delete the unnecessary data's stored in the cloud. Simply archiving the static data file is not suitable for dynamically changing scenarios. The file and the signatures for the file blocks were generated and stored at the server. In this paper we have implemented dynamic data operations like modification and deletion. We are still working on Insertion operation.

5.1. Modification of Data:

If a user wants to replace the particular data block (m_i, i= {1, 2….n}) with the new data mi_{new}. The new tag (tag_{inew}) has been generated using the above said algorithm. Then a new modify message is build consisting of the details.

$Modify_{msg} = \{mi_{new,\ i^{th}\ block,}\ tag_{inew}\}$ (4)

Then it is sent to the server. Server updates the block and generates the proof to the server. The client validates the proof and verifies whether the server has done the modification.

5.2. Deletion of Data:

If a server receives a delete request for a particular block from the user, the server deletes the specified block m_i**.** and then moves the blocks further to generate a new meta data. The client and server interchange the proof of the file F and the Meta data**.**

RESULTS AND DISCUSSIONS

We have carried out this experiment in a real time environment using intel core i5 processor running at 2.5 GHZ, 4 MB RAM. The data is stored in Google Drive for carrying out the test. Our main aim of this paper is to reduce the communication cost incurred in data transfer between client and server.

Size (KB)	0	20	40	60	80	100	120	140	160	180	200
Communication Cost (KB)	245	228	235	250	265	274	280	305	330	340	360

7. Conclusion:

In this paper we propose a method to verify the integrity of the data stored in the cloud server. This helps to keep the individual clients or industries data secure and safe at remote cloud storage. The data integrity is maintained with the help of the cryptographic hash functions and the encryption technique. This protocol also supports dynamic operations like block insertion, deletion and modification. During auditing the content is not leaked to the auditor to ensure privacy. A protocol is implemented to check the data is secure against adversary attacks. It is suitable for thin clients as the data delegation is outsourced in to the auditor. It achieves reasonable communication cost.

REFERENCES

Wang, Q., C. Wang, J. Li, K. Ren and W. Lou, 2009. "Enabling public verifiability and data dynamics for storage security in cloud computing," in Proc. of ESORICS'09, Saint Malo, France: Springer-verlag, pp: 355-370.

Amazon.com, 2009. "Amazon Web Services (AWS)," http://aws.amazon.com.

Ren, K., C. Wang, Q. Wang, 2012. "Security challenges for the public cloud," IEEE Internet Computing, 16(1): 69-73.

Arrington, M., 2006. "Gmail Disaster: Reports of Mass Email Deletions, http://www.techcrunch.com/2006/12/28/gmail-disaster reports of mass-email-deletions.

Ateniese, G., R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson and D. Song, 2007. "Provable data possession at untrusted stores," in Proc. of CCS'07. New York, NY, USA: ACM, pp: 598-609.

Erway, C., A. Kupcu, C. Papamanthou and R. Tamassia, 2009. "Dynamic provable data possession," in Proc. of CCS'09. Chicago, IL, USA: ACM.

Ateniese, G., R.D. Pietro, L.V. Mancini and G.T. Sudik, 2008. "Scalable and efficient provable data possession," in Proc. of SecureComm'08. New York, NY, USA: ACM, pp: 1-10.

Wang Shao-hui, Chang Su-qin, Chen Dan-Wei, Wang Zhi-Wei, "Public auditing for ensuring cloud

data storage security with zero knowledge privacy," International Association for Cryptologic Research, http://eprint.iacr.org/2012/365.pdf.

Dan Boneh, Kevin Lewi, Hart Montgomery, Ananth Raghunathan, 2012. "Key homommorphic and their applications," Crypto 2013[BLM+13], Stanford University.

Wang, Q., C. Wang, K. Ren, W. Lou, J. Li, "Enabling Public Auditability and data dynamics for storage security in cloud computing," IEEE Transactions on Parallel and distributed systems, 22(5): 847-859.

Juels, A. and B.S. Kaliski, Jr., 2007. "Pors: proofs of retrievability for large files," in Proc. of CCS'07. New York, NY, USA: ACM, pp: 584-597.

Sravan Kumar, Ashutosh Saxena, 2011. "Data integrity proofs in cloud storage," in 2011 IEEE conference, ISBN:978-1-4244-8953-4/11/$26.00.