www.ijecs.in

International Journal Of Engineering And Computer Science ISSN:2319-7242
Volume – 5 Issue -02 February, 2016 Page No. 15780-15783

Provable Data Possession at Untrusted Cloud Storage server

Mr. Vitthal Raut[#1]*, Prof. Dr. Suhasini Itkar* [*2]

[#] *M.E. Scholar, Computer Engineering, PES Modern College of Engineering, Pune, India.*[1]
[1]raut.vitthal@gmail.com

[*] *Head of Computer Department, Computer Engineering, PES Modern College of Engineering, Pune, India* [2]
[1]suhasini_naik@yahoo.com

***Abstract*-Cloud computing is an upcoming technology, which offers various services. The services include infrastructure, platform or software services. The service provided by the cloud is over the internet. In the Cloud, the services are available quickly. The cloud has high demand in the market. Most of the organisation or user prefers their storage as on cloud storage and which is located at the remote place. The user has no control over the cloud storage data. The cloud computing uses the resources like Memory, storage and processor, which are not physically present at the user's location, rather they are located outside the premises and managed by a service provider. The user can access the resources via the Internet. The main focus of this paper is to check data integrity of file which is stored on remote cloud storage with less communication overhead.**

Security in terms of integrity is most vital aspects in cloud computing environment. In this paper, we focus on a cloud data security problem. We also, try to get a security with minimal computational overhead because all computation is done over the Internet. The different techniques are discussed for data integrity in cloud storage and their performance is measured in terms of computational overhead.

Keywords— Cloud security, MD5, Integrity, IaaS, PaaS, SaaS.

I. INTRODUCTION

In day to day life, storing data in the cloud has become a trend. A number of clients store their crucial data on cloud servers, without saving a duplicate copy in their local computers. Sometimes data which is stored in the cloud is so important that the clients ensure that it is not lost or corrupted. It is easy to check the integrity of file which is completely downloaded. But it is not a feasible solution to download huge amounts of data just to check data integrity. This consumes a lot of communication bandwidth. It incurs an extra cost of communication. Thus, a lot of works have been done on designing to check remote data integrity which will allow data integrity to be checked without completely downloading the data. The proposed scheme checked the data integrity of remotely located file without downloading it. Provable data possession scheme is designed to check remotely stored data integrity. This scheme has so many benefits over the existing one. We will discuss pros and cons of this scheme in later part of a discussion.

A. What is cloud

Cloud computing simply means “Computing over the internet”. In general cloud computing refers to “computation done through internet”. Due to cloud storage user can access database resources via the internet from anywhere, whenever required. The main benefits of cloud computing is that, the resourced can be provisioned at any time and released when task is completed. The resources might be storage, computational, network or servers.

According to the National Institute of Standard and Technology , US Department of Commerce’s cloud computing is defined as :”on demand network access to pool of shared and configurable resource that can be rapidly provisioned and released with least management effort ” The shared pool of resourced might be networks, servers, storage, applications, and services [1]. Cloud computing is evolved from the technologies like grid computing, distributed computing, parallel computing [1]. Moving user data to cloud storage, security plays vital role. The cloud data should be intact.

According to the University of California, The Department Electrical Engineering and Computer Science (EECS) refer cloud computing as:” the application delivered as services over the Internet and the hardware and systems software in the data center that provides those services”. The cloud provides various services like infrastructure as a service, platform as a service, and software as a service.

The main objective of the study is to design and development of data possession technique on cloud server with minimal overhead. It will help the researchers to recognize security requirements in the various cloud computing service models. This will help to simplify cloud security policies that ensure the security of the cloud environment well-defined.

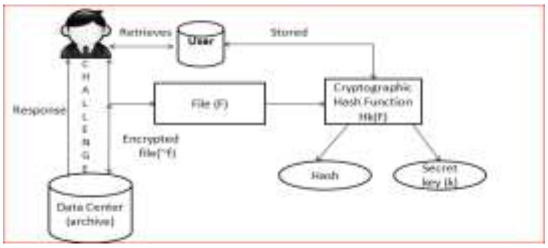
The rest of this paper is organized as follows. In Section 2, data verification techniques are described. In Section 3, the proposed provable data possession presented. In Section 4, complexity analysis and experimental result is discussed. In Section 5, conclusions and possible future work are presented.

II. DATA INTEGRITY VERIFICATION TECHNIQUES

Integrity is the guarantee by which the data is protected from accidental or deliberate (malicious) modification. Hashing techniques, digital signatures and message authentication codes are used to preserve data integrity. Integrity problems are in big scale due to the multi-tenancy characteristic of cloud

A. *Keyed Hash Function:* Hash-based integrity work on hashing technique. This technique contains hash function. This is used to calculate the hash of a file. For each file, it will calculate the unique hash. Hash in unique for each file. The user, before storing the file on cloud calculate the hash of the file using hash function Hk(f), where K is secret key for the hash function. The calculated hash value of the file and stored at is user database. Then the file is stored or moved to the cloud storage. When a user wants to check the integrity of a file, it will access entire file and calculate the hash of a downloaded file (illustrated in Figure 1).

Figure 1 : Hash Based Integrity Technique



This technique is simple and uses hash function. The drawback of this system is to access entire file which is stored on cloud storage. This technique will not feasible where a file is huge in size and need to verify the data integrity of a file. To access entire file will consume a lot of networks bandwidth. By storing multiple hash values for different keys the verifier can check for the integrity of the file F for multiple times, and each one being an independent proof .As this scheme is simple but causes consumption of network bandwidth. For less computational capable devices such as PDA, laptop or mobile this scheme is not suitable.

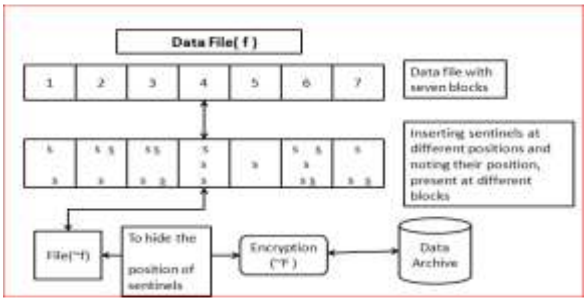
B. Proof of retrievability for large files using sentinels

Ari Juels and Burton S. Kaliski Jr proposed a scheme called Proof of retrievability for large files using sentinels. In this scheme only single key can be used irrespective of the size of the file. Archive needs to access only a small portion of the file F. In this scheme special blocks which are called as sentinel are hidden among other blocks in the data file F [3].

1. *Setup Phase:* In setup phase, this scheme embed sentinel among the entire data block. In each data block sentinel is added at different position. To make sentinel indistinguishable, the whole file is encrypted and stored at cloud server. This scheme involves encryption of whole file F using a secret key K. This scheme also suffers when data to be encrypted is large. This will be computationally cumbersome for small computational devices like PDA, laptop or mobile (illustrated in Figure 2).

.

Figure 2 Proof of retrievability using sentinels


Data File(f)
1
2
3
4
5
6
7
Data file with seven blocks
Inserting sentinels at different positions and noting their position, present at different blocks
File(~f)
To hide the position of sentinels
Encryption (~F)
Data Archive

2. Verification phase: In verification phase owner challenges to the cloud by specifying the position of collection of sentinels and cloud storage to return the associated sentinel values. If the file is modified by then the sentinel position get changed and cloud storage failed to return sentinel position.

This will also causes an additional storage overhead to the cloud storage by adding the newly inserted sentinels. Client also needs to store all the sentinels with it, which will be additional burden to thin client [3].

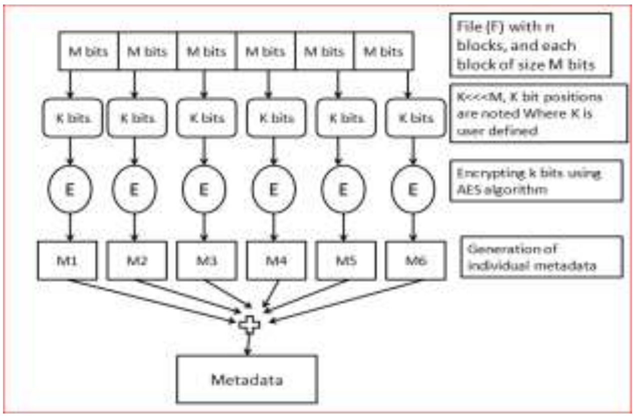
III. Provable Data Possession (PDP):

The proposed protocol gives proof of data possession on remote server. It verifies the integrity of remotely located data. This technique is most suitable for thin client like PDA, mobile, laptop which possesses less computation capability. By using this scheme, user can verify the correctness of data. This proof generally called as proof of data possession (PDP) or Proof of Retrievability (POR) [3].

This technique work in two steps

1) *Setup phase:* In this, technique does not involve the encryption whole data. As illustrated in Figure 3, a file is divided into n block of M bits in each. Among this M bits K bits selected randomly from each block. All selected K bits are encrypted using secrete key k and generate M_1, M_2…M_n metadata. This metadata used in later phase for verification purpose. The main benefit of this scheme is that, it does not allow encryption of whole data. It encrypts only few bits among n bits data. This will drastically reduce computational overhead. Total metadata generated size is n*k bits [2].

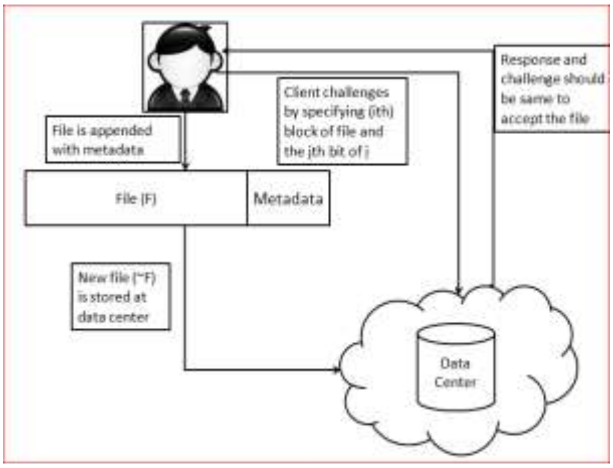
Figure 3 Metadata Generation


M bits
M bits
M bits
M bits
M bits
M bits
File (F) with n blocks, and each block of size M bits
K bits
K bits
K bits
K bits
K bits
K bits
K<<<M, K bit positions are noted Where K is user defined
E
E
E
E
E
E
Encrypting k bits using AES algorithm
M1
M2
M3
M4
M5
M6
Generation of individual metadata
Metadata

2) Verification phase: In Verification phase, a verifier gives the challenge to the cloud server by specifying the <i, j, p> where i is block number, j is bit number and p is the position at which the metadata corresponding the block i is appended.

This metadata will be a k-bit number. Therefore cloud storage server need to send k+1 bit verification metadata. The metadata send by cloud is decrypted by using the number i. The sent metadata and decrypted metadata is compared. If mismatch is found between two means loss of data and security message flow to the cloud user (illustrated in Figure 4).

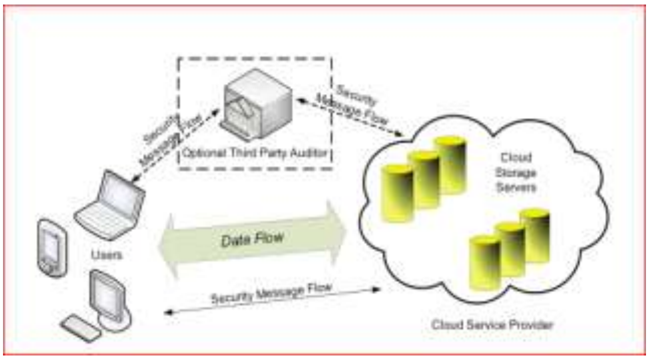
Figure 4 Data Verification


Response and challenge should be same to accept the file
Client challenges by specifying (ith) block of file and the jth bit of i
File is appended with metadata
File (F)
Metadata
New file (~F) is stored at data center
Data Center

A. *System architecture:*

The cloud storage architecture is shown in figure 5; the main component in this architecture is TPA, CSP and Client. A client store data on cloud storage. Cloud storage might be an untrusted. It might hammer the client data and integrity of data is not preserve. To take care this, TPA is introduced, who verify the client data on cloud storage by throwing challenges to CSP, CSP also accept the challenge and through response to the TPA, in this way TPA at regular interval thrown a challenge and CSP gives response. If any mismatch in challenge and response then TPA inform to the client regarding mismatch of data. The client Pre-process the data before storing on cloud, it calculate the metadata which is used for data integrity verification [4] [5].

Figure 5 : Cloud Storage Architecture



B. *Provable data possession schemes*

Consider a file F, consist on n number of blocks: F = (m_1, m_2...m_n). Here m_1, m_2...m_n, represent the fix length bit block. Among this fix length block, we select K bit length metadata respectively from each block. This k bit data from each block is appended to get final metadata. So at the end metadata length is n* k where n is number of block and k is random bit selected from m bit block. The output of the algorithm is represented as follows [8].

X ← A. where | x | is an absolute value of x.

Definition: Provable Data Possession Scheme (PDP): This schema is collection of four polynomial terms (KeyGen, TagBlock, *GenProof, and CheckProof*)

1. *KeyGen* (1^k) → this function generate a secrete key which is used for encryption of metadata. The outcomes of this KeyGen function is Secrete key $S_{k.}$ This is run by TPA to produce secrete key and store at TPA Only.

2. *TagBlock* (s_k, *m*) → *Tm where* T_m is metadata produced by client. This metadata generation is carried out by TPA on the behalf of client; the produced metadata is used for verification purpose [7].

3. *GenProof* ($s_{k,\ F}$, *chal*, Σ) →*V:* This function is used to through authentication challenge to server. It specify the <block number i, j bit number, corresponding metadata position p > to the server and ask corresponding metadata bit. Here Σ is corresponding verification metadata in file F,

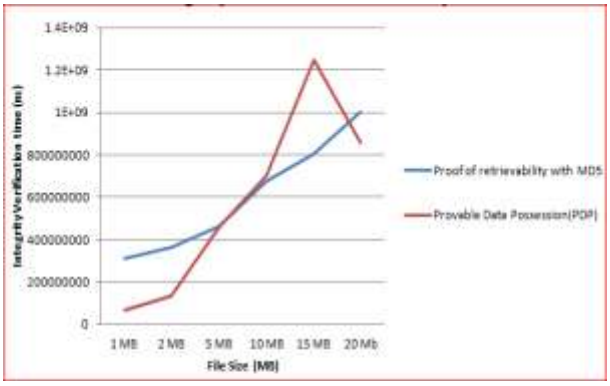
4. *CheckProof* (s_k, *chal, V*) → *success, failure:* is run by the TPA on behalf of client. These validate a proof of data possession. It takes as inputs a secret key s_k , challenge chal and a proof of possession V. This function verifies the corresponding metadata data bit on that basis it through Challenge chal [9].

IV. COMPLEXITY ANALYSIS AND EXPERIMENTAL RESULT

All computations were carried out on an Intel (R) core(TM) I5-4200U CPU @1.60 GHz processor with 4 GB memory running Window 64 bit operating system. We compare PDA scheme with hash based integrity.

In the experiment, we measure the computation costs between PDA scheme and hash based integrity (MD5) when the file length is varied and the block size is fixed. The results are shown in figure 6.

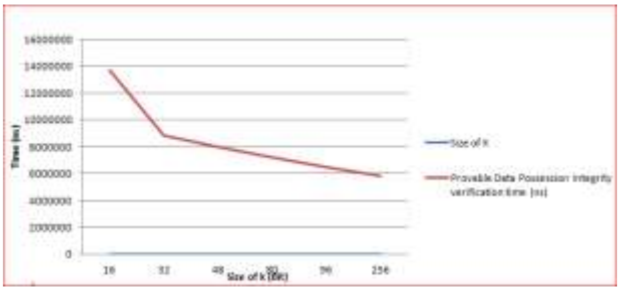
Figure 6 : Integrity Verification Time



The proposed Provable data possession is a probabilistic in nature model; hence the graph shown in figure 5 is not proportional in nature. The selection of block for integrity verification purpose is randomised in nature. We can analyse the result on best case, average case and worst case. In the best case analysis, data modification is found in first attempt and worst case behaviour it found in last attempt. The time complexity in best case is O (1) and worst case it is O (n), where n is number of block in file. In average case, it is O (n/2).

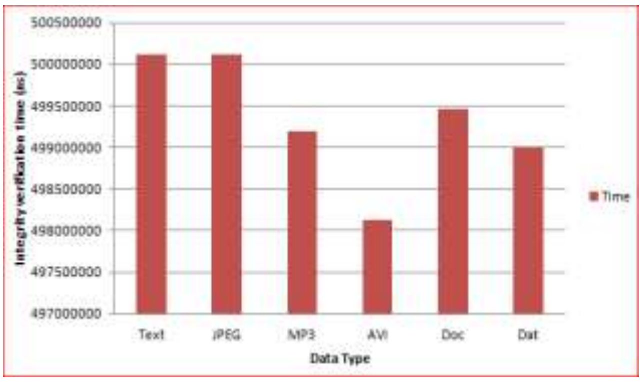
As shown in figure 8 , all the integrity verification time is measured in nano second. All Multiset data of variable size is tested with proposed provable data possession technique. The results are found as shown in figure 8. The time required for integrity verification for all data set is approximately same

Figure 7 K size Vs Integrity Verification



As, shown in figure 7, if the size of k is increased then time required to check the integrity is reduced. Size of k is inversely proportion to integrity verification time. As k size reduced the time required to verify integrity is large and vice versa. As we increased the k size then corresponding metadata generate is more in size. This will cause storage overhead.

Figure 8 Multiset Data Vs Integrity Verification Time



V. Conclusions And Future Work

In this paper, we focussed on the data integrity verification problem, where data is stored on remote cloud storage. We introduced a model provable data possession which verifies the integrity of remotely stored data with minimal communication overhead and computational speed.

This scheme is best suited for small computational devices such as PDA, Laptop, and Mobile. The proposed schemes also reduce the bandwidth utilized for communication purpose. This scheme requires low (or even constant) overhead at the server and requires a small, constant amount of communication per challenge.

It should be noted that this schema is applicable for static data only. It cannot handle the case where data needed to be changed dynamically. Developing such a scheme, which work on dynamic data will be a future challenge

References

[1] Mel P. and Grance G., The NIST Definition of Cloud Computing (Draft), in Proceedings of the National Institute of Standards and Technology, Gaithersburg, pp.6, 2011.

[2] Kaufman L M, (2009)"Data Security in the World of Cloud Computing." IEEE Security and Privacy 7(4):61-64

[3] Cong Wang, Qian Wang, Kui Ren, and Wenjing Lou, "Ensuring Data Storage Security in Cloud Computing", 17th International workshop on Quality of Ser-vice,2009, IWQoS, Charleston, SC, USA, July 13-15,2009, ISBN: 978-1-4244-3875-4, pp.1-9.

[4] Saxena, Sravan Kumar and Ashutosh,"Data Integrity Proofs in Cloud torages", IEEE 2011.

[5] 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.584597.

[6] Bansidhar Joshi, A. Santhana Vijayan, Bineet Kumar Joshi, Securing Cloud computing Environment against DDoS Attacks, IEEE, 2011, pp. 1-5.

[7] E. Mykletun, M. Narasimha, and G. Tsudik, "Authentication and integrity in outsourced databases," Trans.Storage, vol. 2, no. 2, pp. 107138

[8] G.Ateniese, R.Burns, R.Curtmola, J.Herring, L.Kissner, Z. Peterson, and D. Song. "Provable Data possession at untrusted stores. In Proc. of the 14th ACM Conference on Computer and Communications Security (CCS07). ACM Press, October 2007.

[9] Ramgovind S, Eloff MM and Smith E, "The Management of Security in Cloud Computing", IEEE, 2010.

[10] Rohit Bhadauria, Rituparna Chaki, Nabendu Chaki and Sugata Sanyal, A Survey on Security Issues in Cloud Computing, IEEE