International Journal on Future Revolution in Computer Science & Communication Engineering
Volume: 3 Issue: 8
ISSN: 2454-4248
111 – 115

A Dynamic Proxy Oriented Approach for Remote Data Integrity checking along with Secure Deduplication in Cloud

K. Ramya[1], C. Shoba Bindu[2], P. Dileep Kumar Reddy[3]

[1]Department of CSE, JNTUCEA, Ananthapuramu, Andhra Pradesh, India
[2]Department of CSE, JNTUCEA, Ananthapuramu, Andhra Pradesh, India
[3]Department of CSE, JNTUCEA, Ananthapuramu, Andhra Pradesh, India
E- Mail: ramya.kotike@gmail.com[1] , shobabindhu@gmail.com[2], dileepreddy503@gmail.com[3]
K.Ramya: ramya.kotike@gmail.com[1] Tel: +91-9491966798.

Abstract: In Cloud computing users store data over remote servers instead of computer's hard drive. This leads to several security problems since data is out of the control of the user. So, to protect against the security attacks and to preserve the data integrity in the cloud, Huaqun Wang et.al proposed proxy oriented remote data integrity checking (RDIC). However, this scheme only focuses on one-way validation i.e clients have to know whether their files are stored integrally in the cloud. But this scheme does not address the problem of duplication which is essential with increasing demand for cloud storage. And as users are untrusted from the perspective of the server, there is a need to prove the ownership of the files. The proposed work considers the requirement of mutual validation. In this paper we propose a new construction of Identity based RDIC along with secure deduplication. The proposed scheme avoids burden of complex key management and flexible as it support anyone to verify the contents of the data apart from the data owner and incurs less computation cost as token generation is done by the proxy instead of user.

Keywords: *Cloud computing, Identity-based cryptography, Deduplication, Remote data integrity checking.*

*****

I. INTRODUCTION

Cloud computing offers storage and resources according to the demands of the users. With the advent of cloud computing, various benefits can be availed by the user such as he can reduce the hardware and software costs and can access the data irrespective of geographical locations.

However, there are certain barriers before this technique can be widely deployed. In cloud environment, cloud servers are usually not trusted by clients. We consider that cloud servers may tend to access or modify client's data, especially when the files are large and rarely accessed. So it is necessary for clients to know whether their data are stored faithfully in the server.

On the other side, with increasing the popularity of cloud computing, the demand for storage space becomes a critical challenge for cloud storage systems. To manage ever increasing data, deduplication is a well known technique. Data deduplication is a type of data compression technique. It allows eliminating duplicated copies to achieve storage efficiency.

Managing encrypted data storage with deduplication in an efficient way is a practical issue. But current industrial solutions for deduplication cannot handle encrypted data because deduplication and encryption have conflicting strategies. Most of the existing deduplication solutions suffer from brute-force attacks. The problem with deduplication is, computations and communication costs becomes complicated into deduplication process. Second, in an attempt to discover duplicated data, it may affect the privacy of data holders. Third, it may become difficult for the data holder to issue deduplication keys or data access rights to other data holders.

Motivated by the problems stated above we introduce a mechanism based on proxy oriented RDIC along with secure deduplication. We aim to solve the issue of integrity and deduplication of the data in the scenario where the data holder is not available or difficult to get involved. The proposed work reduces the burden of computations done by the user in the process of auditing and deduplication process.

The rest of the paper is organized as follows. Section II gives a brief overview of the related work. Section III presents system and security model. Section IV presents the security and efficiency analysis for the proposed system. At the end, conclusion is provided in section V.

II. RELATED WORK

There exists several storage concerns and problems in cloud computing. With increasing the size of the cloud data, it is impractical to download the complete file for checking the integrity of data as it raises various issues regarding bandwidth cost. Traditional data integrity procedures such as hash functions, authorization code (MAC) cannot be applied directly due to being short of a

 ISSN: 2454-4248
111 – 115

copy of the original file in verification. Blum et al.[1] proposed an auditing issue for the first time that allows to verify the correctness of outsourced data without having knowledge about the entire data . Provable data possession (PDP)[2],[3] at untrusted stores, introduced by Ateniese et al., is a verifiable mechanism based on hash functions and it is a dynamic PDP that allows the data owner to dynamically update the file. There has been many improvisations based on this scheme. Proof of retrievability[4] is a model which depends on processing the data by the client before sending the data or uploading it. Kan Yang et al.[5] proposed another model called as third party auditing which uses an external auditor and homomorphic verifiable tags to reduce communication costs. Yu et al.[6] focused on the problem of the privacy of RDIC protocols for secure cloud storage. But this model is only suitable for the scenario of public key infrastructure (PKI) instead of the identity based framework. Currently, a majority of the existing RDIC constructions rely on PKI where a digital certificate is used to guarantee the genuine of a user's public key. These constructions incur complex key management procedures since certificate generation, certificate storage, certificate update and certificate revocation are time-consuming and expensive.

Deduplication is a storage optimization technique for reducing data redundancy. The idea behind deduplication technique is to identify identical files. Cloud service providers such as Dropbox, Google Drive apply industrial deduplication solutions to minimize storage space. However, they fail in encrypted data deduplication. For example, Some efficient deduplication systems such as DeDu cannot support encrypted data. Most of the clients tend to outsource data in an encrypted format as data is beyond the security premises of the data owner. But applying encryption and data deduplication have conflicting strategies. However, direct deduplication cannot be performed on encrypted data, if clients conventionally encrypt their data. Recently, various PoR schemes were proposed by Wang et.al [9]. However, only few of them put focus on both deduplication and data integrity into account. Hence there is a need to ensure the correctness of the data along with optimizing the storage space.

III. PROPOSED WORK

In this section, we describe the system model and security model of the proposed system. The proposed work is aimed to design a model for integrity verification along with deduplicated storage.

System Model: The Identity based RDIC along with secure deduplication is illustrated in Fig 1. It includes the following entities:

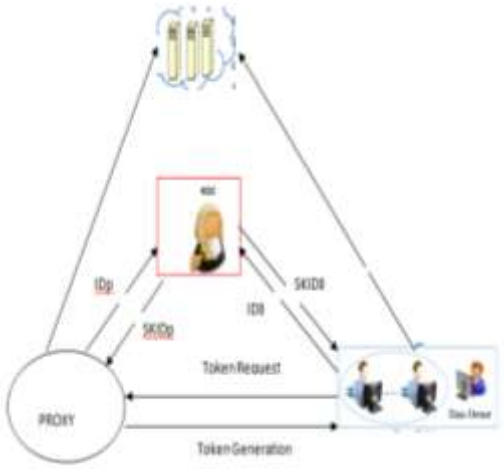

Fig 1: System Model

(i)CSP: An entity that provides computation resources, storage services and to manage the client's data.

(ii)Data holder: An entity that uploads and saves its data in CSP. One or different clients may upload the same data to cloud server.

(iii)Data Owner: The data holder who creates or produces the file is treated as data owner. Priority of data owner is higher than other normal data holders.

(iv)Proxy: An entity which is authorized to process original client's data and is trusted by the data holders to handle data deduplication.

(v)KGC: A trusted entity which can generate the private keys for the corresponding received entities.

The proposed system supporting deduplication and integrity checking includes the following phases:

A.File Uploading Phase: This protocol aims at allowing clients to upload files via the proxy . Specifically, the file uploading phase includes three steps:

Step 1: Before uploading the data, client performs the duplicate check with the public cloud server to know whether such file is stored in cloud storage or not. If duplicate copy exists in cloud server, another protocol called ownership authenticity will be run between the client and the cloud storage server. Otherwise, the following steps (including step 2 and 3) are run between these two entities.

Step 2: client uploads files to the proxy, and receives a receipt from proxy.

 ISSN: 2454-4248
111 – 115

Step3: Proxy helps to generate a set of tags for uploading a file. This tags will be send along with the file to public cloud server.

B. Integrity Checking phase: This phase involves an interactive protocol for integrity verification and it can be run by any entity except public cloud server. In this phase, proxy or client works as the verifier and the cloud server acts as the prover. This procedure includes two steps:

Step1: Set of challenges are generated by a verifier and sends them to the prover. Here verifier can either be the client or the proxy and prover is the cloud server.

Step 2: Based on the stored files and file tags, prover(i.e., cloud server) tries to prove that it exactly owns the target file by sending the proof back to verifier(i.e., cloud client or auditor).

Step3: verifier receives the proof from the prover and outputs true if the integrity verification is passed.

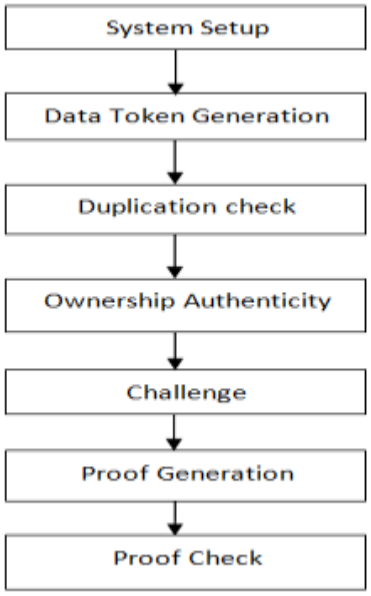

Fig 2: Step by step procedure of proposed work

1.System Setup: It is a probabilistic algorithm run by the KGC. The input to this algorithm is a security parameter k and generates the master secret key and system parameters that are needed for cryptographic operations.

2.Data Token Generation: Data owner first sends its identity to the proxy. Proxy generates data token for the corresponding file and sends it to the user. User will interact with the cloud server and sends the file token.

3.Duplication check: CSP checks if the duplicated data is stored by finding whether token exists. If the check is negative, it requests data upload. User u1 encrypts file F to get cipher text CF. user1 sends CF to CSP, which saves them together with token. If the check is positive and the pre-stored data is from the same user, it informs the user about this situation. If the same data is from a different user, below step will be performed.

4.Ownership Authenticity: User u2 later on tries to save the same data at CSP by following the same procedure described above . That is, u2 first sends token to CSP. Duplication occurs because token already exists. So instead of uploading the entire file CSP challenges the data ownership of user to make sure that the user owns the specified file.

5. Challenge: It is a randomized algorithm run either by the client or the proxy. This algorithm takes input as the system parameters , client's identity ID, and a file name F as input and generates a challenge for the for the file F corresponding to the user ID.

6. Proof Generation: It is a probabilistic algorithm run by the cloud server. It takes the system parameters, the challenge, the data owner's identity ID, the tag t, the file F as input and outputs a proof of the challenged blocks.

7. Proof Check: It will be run by the proxy or the client. It takes the system parameters, the challenge chal, the data owner's identity ID, the file name and a proof as input, outputs 1 or 0 to indicate if the file F keeps intact.

(i)Token Generation Algorithm: To get a file token, the user has to send a request to the proxy. The proxy checks the identity of the user to issue corresponding file token to the client. First, user computes file tag $\Phi f=H(F)$ and sends to the proxy. Then based on the request proxy computes the token $\Phi f,p=TagGen(\Phi f, k)$ which is computed by applying $H_0(H(F), k)$.

Step1: Data owner first sends its identity to the proxy.
Step2: Proxy will find the corresponding privilege P of the user
Step3: User sends file tag Φf= TagGen(F) to proxy
Step4: Proxy computes token $\Phi f,p=TagGen(\Phi f, k)$
Step 5: Send it to the client.
Step 6: Stop.

Fig 4:Token Generation Algorithm

(ii)Duplicate Check Algorithm: Data duplication occurs when a client u tries to upload the same file that has already stored at CSP. To check data duplication, token

 ISSN: 2454-4248
111 – 115

comparison is introduced. If data duplication check is negative, then file has to be uploaded to the CSP. If the data duplication check is positive, then the file has already been existed in cloud server.

Step 1: User sends token to the csp.
Step 2: If duplication check is negative
Step 2.1: User computes the encrypt file CF=Enc(Kf,F)
Step 2.2: Uploads CF to the CSP along with token
Step 3: Else
Step 4: If duplicate check is positive
Step4.1: User needs to run proof of ownership protocol with csp
Step 4.2: User sends request to the proxy for token generation
Step 4.3: Proxy computes Φf,p=TagGen(Φf, k) which will be send to the csp
Step 5: End

Fig 3: Duplicate check Algorithm

B: Remote Data Integrity Checking This checking is aimed to provide the integrity of client's data against misbehaviour of servers. In the integrity auditing protocol, either the proxy or the client works as the verifier. The integrity protocol is designed in a challenge response model. Specifically, the verifier randomly picks a set of block identifiers and ask the cloud server. The verifier can either be the client or the delegated proxy. The procedure happens in the following steps:

1. KGC generates master secret key x. Then it computes $y=g^x$
2. Input the original client ID_0, KGC computes $SKID_0=(R0, \sigma_0)$
3. Then KGC sends $SKID_0$ to the original client.
4. Similarly, input the proxy's identity ID_p, the proxy can also get its private key $SKID_p=(R_p, \sigma_p)$.
5. On receiving the warrant signature pair proxy computes the secret key , σ_1
6. A 2-move interactive protocol takes place between client and PCS using (R_o, R_p).

This is the verification phase performed by the verifier using challenge response protocol. It is an interactive proof system between prover and verifier. A randomized algorithm will be run by the proxy to generate the challenge and this challenge is send to CSP. Upon receiving challenge, the cloud server generates proof for the challenged blocks. At the end of the proof client outputs 0 or 1 indicating success or failure.

IV. PERFORMANCE ANALYSIS

A. Security Analysis: This section analyzes the security performance of the proposed scheme.

Unforgeability: The protocol satisfies the security property of unforgeability. The private key that was extracted for the corresponding identity is (R,σ). Here only part of the private key is specified to other entities. Though private key R is made public, its publicity has no provision to leak the other part of the private key σ and hence it is existentially unforgeable. Hence protocol leaks no information of the stored file to the verifier.

Brute-Force Attack: security is thus only possible when a message is unpredictable. The primitive methods that are used in traditional convergent encryption to support duplication check derives a key from the file F by applying cryptographic hash function K=H(F). To avoid brute force attack, the key will be generated with the help of the proxy with privilege key k_p. In this way, both the proxy and CSP cannot have a chance to decrypt the cipher text.

B. Efficiency Analysis: In the experiment, the performance is measured from perspective of computation costs and memory utilization.

(i)Computation Cost Efficiency: Here we analyze the computation cost of client and PCS.

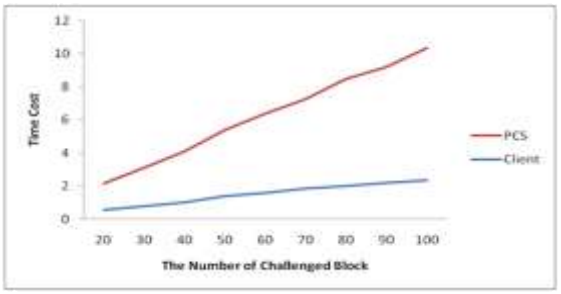

Fig 5: Computation cost of client and PCS

Fig 5 depicts the computation cost of PCS and client. X-label denotes the time cost (second) and y label denotes the challenged blocks. From the figure it is evident that most of the computation is done by the PCS. Client performs little computation and hence suitable for moblile phones where computation resource is limited.

(ii)Storage Efficiency: In this section, we measure the memory utilization before and after deduplication

 ISSN: 2454-4248
111 – 115

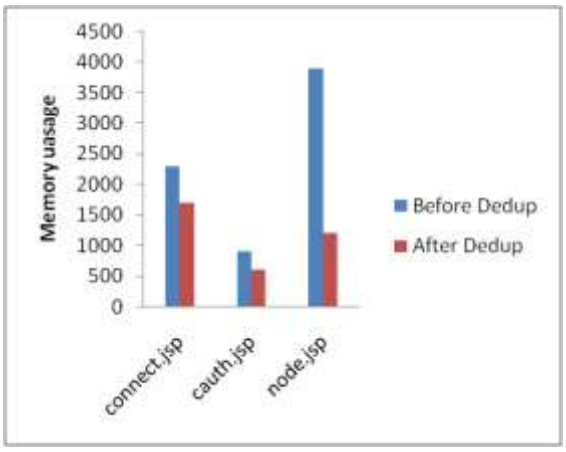

Fig 6: The comparison of memory utilization

Fig 6 compares the memory utilization before performing deduplication technique and after performing deduplication technique. The given figure demonstrates that the requirement of storage space is more before performing deduplication technique and is less after performing deduplication technique. Connect.jsp, cauth.jsp, node.jsp are unique files which are uploaded into the cloud. The time taken to upload and encrypt the duplicate files is skipped. Hence total time spent on uploading a file is greatly reduced.

V. CONCLUSION

Secure deduplication is an effective technique for minimizing cloud storage space while preserving the security and privacy of cloud data. In this paper we discussed the data integrity checking along with deduplication. We formalized a system model and its analysis show that its computation cost is minimum because the token generation is done by the proxy instead of user. So, the proposed protocol is efficient for both the cloud server and the verifier.

REFERENCES

[1] M. Blum, W. Evans, P.Gemmell, S. Kannan, M. Naor, "*Checking the correctness of memories*". Proc. of the 32nd Anual Symposium on Foundations for computers, SFCS 1991, pp. 90–99, 1991.

[2] G. Ateniese, R. C. Burns, R. Curtmola, J. Herring, L. Kissner, Z. N. J. Peterson, D. X. Song, "*Provable data possession at untrusted stores*". ACM Conference on Computer and Communications Security, 598-609, 2007.

[3] G. Ateniese, R. C. Burns, R. Curtmola, J. Herring, O. Khan, L. Kissner, Z. N. J. Peterson, and D. Song, "*Remote data checking using provable data possession*". ACM Trans Inf. Syst. Secur., 14, 1–34, 2011.

[4] A. Juels and B. S. K. Jr., "*Pors: proofs of retrievability for large files*," in ACMConference on Computer and Communications Security, P. Ning, S. D. C. di Vimercati, and P. F. Syverson, Eds.ACM, 2007, pp. 584– 597.

[5] Kan Yang and Xiaohua Jia.,"*TSAS: Third-Party Storage Auditing Service* in Security for Cloud Storage Systems",SpringerBriefs in Computer Science 2014, pp 7-37

[6]Y. Yu, M H Au, Y. Mu, S. Tang, J. Ren, W. Susilo, and L. Dong, "*Enhanced privacy of a remote data integrity checking protocol for secure cloud storage*", International Journal of Information Security, 14(4): 307– 318, 2015.

[7] J. Li, Y. K. Li, X. F. Chen, P. P. C. Lee, and W. J. Lou, "*A hybrid cloud approach for secure authorized deduplication*," IEEE Trans. Parallel Distrib. Syst., vol. 26, no. 5, pp. 1206–1216, May 2015, doi:10.1109/TPDS.2014.2318320.

[8] Huaqun Wang, Debiao He, Shaohua Tang," *Identity-Based Proxy-Oriented Data Uploading and Remote Data Integrity Checking in Public Cloud*" IEEE Transactions on Information Forensics and Security, vol..11, No: 6, June 2016.