Journal of Advances in Science and Technology
Vol. 12, Issue No. 25, (Special Issue) December-2016, ISSN 2230-9659

Review on Data Deduplication and Integrity Auditing in Cloud

Lakshmi Hunashikatti[1]*, Prof. P. M. Pujar[2]

[1]Dept. of Computer Science and Engineering, Gogte Institute of Technology, Belagavi, India

[2]Dept. of Computer Science and Engineering, Gogte Institute of Technology, Belagavi, India

Abstract – With the boundless advancement in cloud computing technology throughout late years, storing of huge data to cloud has turned out to be an attractive trend; with this the customer can easily store and maintain information. The fact of the matter is to propose two safe systems in particular SecCloud and SecCloud+ which can accomplish both deduplication and secure auditing of data in cloud. The SecCloud structure includes auditor which helps in dividing the record into blocks, each block is doled out an advanced mark and is assigned a digital signature and is checked for its integrity before uploading it on the cloud. The configuration of SecCloud+ empowers secure deduplication and information auditing on encoded information as clients dependably need to scramble their information before exchanging to preserve file confidentiality.

Keywords— Integrity Auditing, Deduplication, SecCloud, SecCloud+, Digital signature.

----------♦----------

1. INTRODUCTION

The immense development in cloud computing area provides the clients a wide range of benefits such as data accessibility from geographically distributed areas, reliability, rapid deployment, data archival, disaster recovery and strong protection for backup. Despite of immense benefits provided by the cloud storage there are some serious security concerns like “Integrity auditing” and “Deduplication”. In this paper we study the above issues and propose the design of two secure frameworks namely “SecCloud” and “SecCloud+”. Section II gives the detailed study of the Security concerns and related work in the area. In Section III we study about the System Architecture. Section IV describes about the Implementation of the proposed secure systems. Section V draws the Conclusion of the paper.

2. BACKGROUND THEORY

A. Integrity Auditing

Data integrity means that the data must be correctly stored on the cloud storage without any modifications and must detect for any violations i.e. data lost, altered or compromised. As the data is transferred over the network in cloud storage model unlike in traditional data storage systems, the data may get altered or corrupted and also is stored in an uncertain (untrusted) domain, on which the user doesn’t have control; this leads to major concern for security and integrity. This issue can be summed up as [7] “*how effectively can the client perform integrity verifications occasionally without the local copy of data files.*”

The author in their work [1], [2] introduced the definition of PDP (provable data possession). PDP allows a client who has stored the data at an untrusted server to verify the possession of a file at server without actually downloading the file. The verifier can perform the verification with only a small amount of metadata (tags) without actually downloading the files (blocks). The author further improved the PDP by including dynamic scenarios. With the introduction of dynamic provable data possession [DPDP] the author in this paper [3] improvised the work of [1] this extends the basic PDP model [1] to support dynamic scenarios with insertions by the use of authenticated flip tables.

Another work supporting integrity auditing and in line with PDP is (POR) proof of retrievability [5]. Compared to PDP, POR along with the assurance of possession of targeted file with the cloud server also ensures the retrievability of file from server. Author in paper [15] proposes two schemes one for public verifiability and another scheme allows only private verification where both the schemes rely on the homomorphic signatures to generate a proof into one small authenticator value. In this paper [6] the author tried to combine public verifiability and dynamic data operations together in his work by improving the POR model in which the typical construction of Merkle

hash tree which is used for block tag authentication is manipulated.

B. Secure Deduplication

The secondary issue confronted is secure deduplication. It is a data compression strategy to reduce storage needs by eliminating redundant data, which helps to save the network bandwidth and storage space of cloud servers. The second issue can be summed up as [7] "*in what capacities can the cloud servers affirm that the customer possesses the transferred record before making a connection to this document*"

The notion of "(PoW) proof of ownership" was introduced in [9] which allow a client to demonstrate to the prover (may be auditor or server) that it claims the targeted document. To enable secure "client side deduplication" arrangements taking into account Merkle tree and particular encoding were displayed. The authors in [9] addressed one of the most severe security risks: an adversary claiming to possess a file that has only a fraction of the original file.

The above stated works either consider secure deduplication or integrity auditing". In the proposed work an attempt is made to simultaneously solve both the problems. This proposed work is different from [3] which also audits integrity while achieving deduplication because of following reasons:

1. The hash tag generation which was a load on client is now moved on to the auditor.

2. It audits the integrity and performs deduplication check on the encrypted data.

3. PROBLEM DEFINITION

The rising cloud computing technology developed during past ten years is turning into an alluring pattern for outsourcing the information to be put away in cloud. This outsourcing of data reduces the efforts of data management and maintenance on users. As the data is transferred over the network it is not trust worthy to use the cloud servers as it involves security risks with realization of data deduplication in cloud while achieving integrity auditing.

C. Problem Statement

To design and implement two secure systems SecCloud and SecCloud+ in order achieve data integrity and deduplication in cloud.

D. Objectives

The secure systems SecCloud and SecCloud+ are designed to achieve the following properties.

1. **Integrity Auditing:** It refers to the verification of accuracy and consistency of the remotely stored data.

2. **Secure Deduplication:** It requires elimination of redundant data in cloud server by and keeping single copy of the same data.

3. **File Confidentiality:** The aim of file confidentiality requires preserving the privacy of the users by encrypting the contents of a file.

4. SYSTEM ARCHITECTURE

The architecture of the Secure Systems involves the below entities as indicated in Figure 1.

Figure1. System Architecture of the proposed system

1. Cloud Clients: They are the users who can be individuals or corporate clients. These clients can store volumes of data to the cloud server and rely upon the cloud for efficient data management and maintenance.

2. Cloud Servers: The cloud clients can buy or rent storage space on cloud provided by the cloud hosting company. A cloud server is such a rented space which virtualizes the physical storage and other resources as per the user's requirements to expose them as storage pools.

3. Auditor: The Cloud clients store the data on cloud server, this data needs to be checked for correctness at regular intervals and also avoid redundant data. To achieve this Auditor is used which acts like a certificate authority. As it certifies the movement of data the auditor is associated with a pair of public and private keys. The public key is made available to the other clients or entities in the system. Auditor performs Integrity auditing and deduplication check.

The system uses three protocols to achieve auditing and deduplication

1. Proof of Ownership Protocol: between Cloud Client and Server

2. File Uploading Protocol: between Client and Auditor or Auditor and Server

3. Integrity Auditing Protocol: between Client/Auditor and Server

5. IMPLEMENTATION

The proposed system involves implementation of two secure systems namely SecCloud and SecCloud+.

E. Module 1: SecCloud

The SecCloud system supporting file-level deduplication includes the following three protocols respectively as in Fig.5.1.

1) *File Uploading Protocol:* The protocol goes for permitting customers to transfer documents by means of the auditor. In particular, the file uploading protocol incorporates three stages:

- *Stage 1* (cloud customer→ cloud server): client performs the copy check with the cloud server to affirm if such a document is archived in cloud storage before moving a file on to server. In the event that there is a copy, another convention called Proof of Ownership will be keep running between the Cloud client and the distributed storage server. Something else, then accompanying steps (phase 2 and phase 3) are kept running between these two entities.

- *Stage 2* (cloud client → auditor): customer (Cloud Client) transfers files to the examiner (auditor), and gets a receipt from auditor.

- *Stage 3* (auditor → cloud server): Auditor now divides the file into blocks, hash tags are generated for each of the blocks and a signature is created for individual blocks. The signature of all the blocks is combined and then concatenated with the message to form a Digital signature which proves the ownership of the file possessed

2) *Integrity Auditing Protocol:* This is an interactive protocol used for information trustworthiness (integrity) verification and permitted to be introduced by any entity aside from the cloud server. In this convention, the cloud server assumes the part of prover, while the examiner or customer acts as the verifier. There are two stages in this convention:

- *Stage 1*(Cloud client/auditor → server): verifier (Cloud customer or auditor) produces an arrangement of difficulties and sends them to the prover (cloud server).

- *Stage 2* (server → customer/reviewer [Auditor]): in view of the archived files and hash tag of file blocks, prover (cloud server) tries to demonstrate that it precisely claims the targeted file by sending the confirmation back to verifier (cloud customer or auditor). Towards the end of this convention, verifier yields genuine if the trustworthiness confirmation (Data integrity verification) is passed.

3) *Proof of Ownership Protocol:* It is an intuitive protocol introduced at the cloud server for checking customer precisely possesses a guaranteed file. This convention is regularly activated along with the File Uploading protocol to keep the spillage of side channel data.On the difference to the integrity auditing protocol, in PoW the cloud server works as verifier, while the client assumes role of the prover. This protocol additionally incorporates two stages

- *Stage 1*(cloud server → customer): cloud server creates an arrangement of difficulties and sends them to the customer (Cloud client).

- *Stage 2*(cloud client → server): the customer reacts with the verification for file possession, and cloud server at last confirms the legitimacy of confirmation.

F. Module 2: SecCloud+

The SecCloud+ includes an extra trusted entity, to be specific key server. This is in charge of allocating customers with secret key (as indicated by the file content) for encoding documents. This engineering is in accordance with the late work [4]. However, the proposed work is distinguished from the past work [4] by taking into consideration of integrity auditing on encoded information. SecCloud+ utilizes the same three protocols (the file uploading protocol, the integrity auditing protocol and the proof of ownership protocol) as with SecCloud. The main difference is the file uploading protocol in SecCloud+ includes an extra step for correspondence between cloud customer and key server. The document to be uploaded is checked for deduplication and afterwards it is encrypted and marked by the owner of the record. Presently the SecCloud+ has encrypted documents unlike to in SecCloud which archives plain files.

G. Algorithm / Technique Used

1) Elliptic Curve Cryptography ***(ECC)***

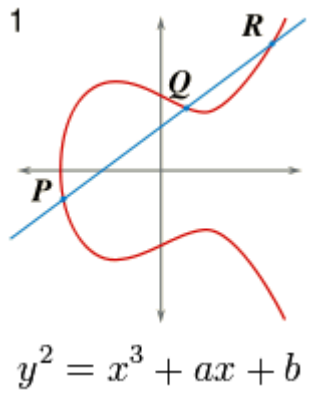

Figure2. Simple Elliptic Curve

Victor Miller (IBM) [8] and Neil Koblitz (University of Washington) [9] discovered Elliptic Curve Cryptography (ECC) in 1985 as second approach to existing public-key cryptographic systems (RSA and Diffie Hellman). ECC makes use of algebraic structures of the elliptic curves over finite fields. Basically ECC follows addition of rational points on a chosen elliptic curve. Suppose if following are the parameters then the Elliptic curve, points on the curve and the equation for the curve can be represented as shown in Figure 2

E -> Chosen Elliptic Curve

P -> Point on the Elliptic curve

n -> Maximum limit to be chosen (This should be prime number)

Elliptic curve cryptosystems incorporates key distribution, encryption and decryption algorithms. In the key distribution algorithm the clients share a secret key. Key generation is a crucial part as it includes the generation of public key and private key. These keys are used by the sender and receiver, where in the sender encrypts the message with the receiver's public key and the receiver decrypts it with his own private key.

The ECC domain parameters of an Elliptic curve EC over field F_p is defined by the tuple as given below

D = (*q, FR, a, b, G, n, h*), where

- *q*: prime power, that is q = p or q = 2^m, where p is a prime
- *FR*: field representation of the method used for representing field elements $\in F_q$
- *a, b*: field elements, they specify the equation of the elliptic curve E over F_q, $y^2 = x^3 + ax + b$
- *G*: A base point represented by G= (x_g, y_g) on E (F_q)
- *n*: Order of point G , that is n is the smallest positive integer such that nG = ***O***
- *h*: cofactor, and is equal to the ratio #E(F_q)/n, where #E(F_q) is the curve order

2) Algorithm for Encryption

To Encrypt a plain text, following steps are performed

Step 1: Select a random integer d which is in the range[1 - n-1], n is the order of the curve selected.

Step 2: Calculate Q =d* G, where G is the base point picked before.

Step 3: Private KeySk1 and public key Pk1 are assigned as below

Step 4: The text message is converted into bit string and divided into chunks for generating encrypted file. Each block of file is encrypted in the form of (Chosen point, Encoded Point).

Step 5: This Encrypted text along with the points on curve is sent to receiver.

3) Algorithm for Decryption

To decrypt a cipher text, following steps are performed

Step 1: Sender selects a random integer d that is between 1 to n-1, where n is the order of the curve selected.

Step 2: Calculate Q = d* G, G is the base point chosen earlier.

Step 3: The receiver also performs the calculation for choosing the private key Pr2 and public key Pu2.

Step 4: The decryption is performed by the received encrypted text, as the encrypted text is received in chunks each chunk of file is decrypted one by one as messages [i]. Encoded Point - Pr2* messages [i]. Chosen Point

Step 5: Then the result is got as bit strings, which are later converted into respective alphabet or number.

Step 6: And finally the sent message is retrieved by the encrypted text.

4) Digital Signature Generation

Files stored in the cloud can be deleted by either the group manager or the data owner (i.e., the member who uploaded the file into the server). To delete a file

IDdata, the auditor computes a signature μf1 (IDdata) and sends the signature along with IDdata to the cloud server. The server will delete the file if the equation e (μf1 (IDdata), P) =e (W, f1 (IDdata) holds

Signature Generation Algorithm

Input: public key (A, B,h), system parameters, message m

Output: Generate a valid signature on M.

Begin

Select random numbers a, roM, roR, mus, mux, mueprime, mut, muE

Computes the following values

E0 = g * roE

E1 = h+ (h1*roE)

E2 = h+ (h2*roE)

ACOM (A*(a2^rom) mod n).mod n

s = (Eprime + ke)* roM

BCOM (B *(w^ roR mod l) mod l

t = Eprime * roR

V0 = g*muE

V1 = (g * mux+ (h1 * muE)

V2 = (g*mux)) + (h2* muE)

Vmpk = (((a1^mux mod n)*(a2^mus mod n)).mod (n))*(ACOM^~mueprime mod n)).mod (n));

Vrev = ((w^mut mod l))*(BCOM^~mueprime mod l))) mod l

E = E0+ E1+E2

V = V0+V1+V2

reste = ACOM+ BCOM+ V + Vmpk+ Vrev

Set c = f (E + reste + message)

Construct the following numbers

taux = c * (x +mux)

taus = c * (s+mus)

taut = c *(t+mut)

tauePrime = c *(Eprime + mueprime)

tauE = (c*(roE+muE)) mod(o)

Return

σ = (E0, E1, E2, ACOM, BCOM,c, taux, taus, taue Prime, taut)

End

5) Signature Verification Algorithm

Input: system parameters and signature σ = (E0,E1,E2,ACOM,BCOM,c,taux,taus,tauePrime,taut,t auE)

Output: True or False.

Begin

Compute the following values

taue = (c*(expKe+tauePrime);

tauEG = g *tauE

a0a1 = (a0^c mod n)*(a1 ^ taux mod n)) mod n

a2A= (a2^taus modn)*(ACOM^ ~taue mod n) mod n

Vmpk = (a0a1 * a2A) mod n

Bw= ((b^c mod l)*(w^taut mod l)) mod l

Vrev = (bw*(BCOM^~tauePrime mod l)) mod l

E = E0+ E1+E2

V = V0+V1+V2

reste = ACOM+ BCOM+ V + Vmpk+ Vrev

If c = f (E + reste + message)

Return True

Else

Return False

End

6. CONCLUSION

The aim is to implement two secure frameworks SecCloud and SecCloud+ which accomplish both "integrity auditing and deduplication in cloud". To achieve this the SecCloud subsystem presents an auditing entity, which helps client produce hash tags for the block of data before transferring the file to server and also checks regularly the correctness of data in the server. The configuration of SecCloud+ empowers secure deduplication and information auditing on encoded information as clients dependably need to encrypt their information before transferring to the server in order to preserve file confidentiality.

ACKNOWLEDGMENT

It is a great for the author to acknowledge the assistance and contribution of a large number of individuals towards this effort. We would like to express our deep appreciation to all the members of our institution (GIT), friends and family members for their support and much needed encouragement.

REFERENCES

G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, (2007). "Provable data possession at untrusted stores," in *Proceedings of the 14th ACM Conference on Computer and Communications Security*, ser. CCS. New York, NY, USA: ACM, 2007, pp. 598–609.

G. Ateniese, R. Burns, R. Curtmola, J. Herring, O. Khan, L. Kissner, Z. Peterson, and D. Song (2011). "Remote data checking using provable data possession," *ACM Trans. Inf. Syst. Secur.*, vol. 14, no. 1, pp. 12:1–12:34.

C. Erway, A. K¨upc¸ ¨u, C. Papamanthou, and R. Tamassia (2009). "Dynamic provable data possession," in *Proceedings of the 16th ACM Conference on Computer and Communications Security*, ser. CCS '09. New York, NY, USA: ACM, pp. 213–222.

S. Keelveedhi, M. Bellare, and T. Ristenpart (2013). "Dupless: Server aided encryption for deduplicated storage," in *Proceedings of the 22Nd USENIX Conference on Security*, ser. SEC. Washington, D.C.: USENIX Association, 2013, pp. 179–194.

H. Shacham and B. Waters (2008). "Compact proofs of retrievability," in Proceedings of the 14th International Conference on the Theory and Application of Cryptology and Information Security: Advances in Cryptology, ser. ASIACRYPT. Springer Berlin Heidelberg, pp. 90–107.

Q. Wang, C. Wang, J. Li, K. Ren, and W. Lou, (2009). "Enabling public verifiability and data dynamics for storage security in cloud computing," in Computer Security – ESORICS, M. Backes and P. Ning, Eds., vol. 5789. Springer Berlin Heidelberg, pp. 355–370.

Jingwei Li, Jin Li, DongqingXie and Zhang Cai (2015). "Secure Auditing and Deduplicating Data in Cloud" DOI10.1109/TC.2015.2389960, IEEE Trans 2015

N. Koblitz (1987). Elliptic Curve Cryptosystems, *Mathematics of Computation,* volA8.

V. S. Miller, "Use of Elliptic Curves in Cryptography". Advances in Cryptology CRYPTO, New York, Springer - Verlag

Corresponding Author

Lakshmi Hunashikatti*

Dept. of Computer Science and Engineering, Gogte Institute of Technology, Belagavi, India

E-Mail – lakshmi.hunashikatti@gmail.com