Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X
IMPACT FACTOR: 5.258

IJCSMC, Vol. 5, Issue. 4, April 2016, pg.397 – 402

Prevention and Verification of Data Deduplicating and Integrity Auditing in Cloud

Dr. Mohammed Abdul Waheed[1], Shahnawaz Parveen[2]

[1]Associate Professor, [2]P G Student

Department of Computer Science and Engineering, VTU Centre for PG studies, Regional Office, Kalaburagi, Karnataka, India

[1] dr.mawaheed@gmail.com; [2] shahnawazparveen84@gmail.com

Abstract— The advent of the cloud computing makes storage outsourcing a rising trend, which promotes the secure remote data auditing and data deduplication. In this work, we study the problem of integrity auditing and secure deduplication on cloud data. Specifically, aiming at achieving both data integrity and deduplication in cloud, we propose two secure systems, namely SecCloud and SecCloud+. SecCloud introduces an auditing entity with a maintenance of a MapReduce cloud, which helps clients generate data tags before uploading as well as audit the integrity of data having been stored in cloud. Compared with previous work, the computation by user in SecCloud is greatly reduced during the file uploading and auditing phases. SecCloud+ is designed motivated by the fact that customers always want to encrypt their data before uploading, and enables integrity auditing and secure deduplication on encrypted data

Keywords— Cloud Computing, Integrity Auditing, Data Deduplication, cloud security , Intelligent Compression

I. INTRODUCTION

Even though cloud storage has been widely used adopted, it fails to accommodate some important emerging needs such as the abilities of auditing integrity of cloud files by cloud clients and detecting duplicated files by cloud servers. We disclose both problems.

The first problem is integrity auditing. Data integrity demands maintaining and assuring the accuracy and completeness of data. A data owner always expects that his data in a cloud can be stored correctly and trustworthily. It means that the data should not be illegally tampered, improperly modified, deliberately deleted, or maliciously fabricated. If any undesirable operations corrupt or delete the data, the owner should be able to detect the corruption or loss. Further, when a portion of the outsourced data is corrupted or lost, it should still be retrieved by the data users [1].

The second problem is secure deduplication. Data stored at remote cloud servers are often duplicated. This fact raises a technology named deduplication, in which the cloud servers would like to deduplicate by keeping only a single copy for each file and make a link to the file for every client who owns or asks to store same file[3][2]. It is generalized to how can the cloud server efficiently confirms that the client owns the uploaded file before creating a link to this file for him/her.

The challenge is to achieve data integrity and deduplication in cloud by proposing two secure systems namely SecCloud and SecCloud+. SecCloud introduces an auditing entity with maintenance of a Map Reduce cloud, which helps clients generate data tags before uploading as well as audit the integrity of data having been stored in cloud. This design fixes the issue of previous work that the computational load at user or auditor is too huge for tag generation. Motivated by the fact that customers always want to encrypt their data before uploading, for reasons ranging from personal privacy to corporate policy, we introduce a key server into SecCloud as with and propose the SecCloud+ schema. Besides supporting integrity auditing and secure deduplication, SecCloud+ enables the guarantee of file confidentiality.

II. RELATED WORK

In this section, we present literatures of several highly related research areas to integrity auditing and data deduplication.

A. *Integrity Auditing*

Provable Data Possession (PDP) was introduced by [5][6] for assuring that the cloud servers possess the target files without retrieving or downloading the whole data. Many works were concerned on how to realize PDP on dynamic scenario: [7] proposed a dynamic PDP schema but without insertion operation; and this was improved by introducing authenticated flip table by Erway et al. [8]. Proposals of [5][7][8][9] suffered from the computational overhead for tag generation at the client. To fix this issue, [10] proposed proxy PDP in public clouds and [11] proposed the cooperative PDP in multi-cloud storage.

Integrity auditing is also supported by Proof of Retrievability (POR) [12]. Compared with PDP, POR assures the cloud servers possess the target files, and guarantees their full recovery. In [12], clients apply erasure codes and generate authenticators for each block for verifiability and retrievability. To achieve efficient data dynamics, [13] improved the POR model by manipulating the classic Merkle hash tree construction for block tag authentication.[16] combined the privacy preserving word search algorithm with the insertion in data segments of randomly generated short bit sequences and developed a new POR protocol.

B. *Secure Deduplication*

Deduplication is a technique where the server stores only a single copy of each file, regardless of how many clients asked to store that file, such that the disk space of cloud servers as well as network bandwidth are saved. However, trivial client side deduplication leads to the leakage of side channel information. For example, a server telling a client that it need not send the file reveals that some other client has the exact same file, which could be sensitive information in some case. In order to restrict the leakage of side channel information,

The Proof of Ownership protocol was introduced by [3] which lets a client efficiently prove to a server that the client exactly holds this file. To enable secure client-side deduplication several proof of ownership protocols based on the Merkle hash tree are proposed [3]. Pietro and Sorniotti [19] proposed an efficient proof of ownership scheme by choosing the projection of a file onto some randomly selected bit-positions as the file proof.

Data deduplication focuses on the confidentiality of deduplicated data and considers to make deduplication on encrypted data. [20] introduced the private data deduplication as a complement of public data deduplication protocols of [3]. For ensuring data privacy in deduplication Convergent encryption [21] is a promising cryptographic primitive.

III.PURPOSE

The main aim of the project is to achieve integrity auditing and deduplication on plain data and encrypted data stored on cloud servers.

A. *Existing System*

Cloud security is an important factor for the users of cloud. Data stored at cloud server may suffer from deduplication and integrity auditing. To achieve integrity auditing Proxy PDP in public cloud was proposed by [10] and POR (Proof of Retrievability) assures cloud servers posses the target files and guarantees their full recovery. In [12], clients apply erasure codes and generate authenticators for each block for verifiability and retrievability. Data deduplication was achieved as a complement of public data deduplication protocols of [3] allowing client side deduplication to identify deduplication opportunities already at the client and save the bandwidth of uploading copies of files to server requiring client to generate tag.

B. *Disadvantages of Existing System*

- Client is responsible for tag generation during auditing.
- Client side deduplication leads to leakage of side channel information.
- This action of deduplication would lead to a number of threats potentially affecting the storage system.

C. *Proposed System*

In this article, we aiming to achieve data integrity and deduplication in cloud by proposing two secure systems namely SecCloud and SecCloud+. SecCloud is introducing an auditing entity with maintenance of a MapReduce cloud, that is helping clients generate data tags before uploading files as well as audit the integrity of data having been stored in cloud. SecCloud+ supports the guarantee of file confidentiality besides supporting integrity auditing and secure deduplication, SecCloud+ is designed for the fact that customers always want to encrypt their data before uploading, and thereby enabling integrity auditing and secure deduplication on encrypted data.

D. *Advantages of Proposed System*

- Security considered in SecCloud is the prevention of leakage of side channel information.
- It follows proof of ownership protocol between clients and servers which allow clients to prove to cloud server that they exactly own the target data.
- It achieves deduplication while preventing Dictionary attack.
- The client is relived from tag generation as it is now the task of Auditor.
- Project can be implemented with real time cloud.
- It is user friendly.
- It is portable and authentication can still be improved.

IV. IMPLEMENTATION

System models for SecCloud and SecCloud+ both follow the same architecture. We have three entities serving as three modules:

Cloud Client: For data maintenance and computation Cloud Clients have large data files to be stored on the cloud. They can be either individual consumer or commercial organization. They can upload files in SecCloud and SecCloud+ models. They perform integrity auditing and deduplicaton of data stored on cloud servers.

Cloud Servers: Cloud servers mean virtual servers which run on cloud computing environment. They Virtualizes the resources according to the requirements of clients and expose them as storage pools. The cloud clients may buy or lease storage capacity from cloud server and store their individual data in these bought or rented spaces for future utilization. They store data in SecCloud or SecCloud+ models.

Auditor: Auditor offers its auditing service with more powerful computation and communication abilities helping clients upload and maintain a MapReduce cloud and acts as a certificate authority. The users may resort to Auditor for ensuring the storage security of their outsourced data.

System models for SecCloud and SecCloud+ both follow the same architecture as shown in figure 1

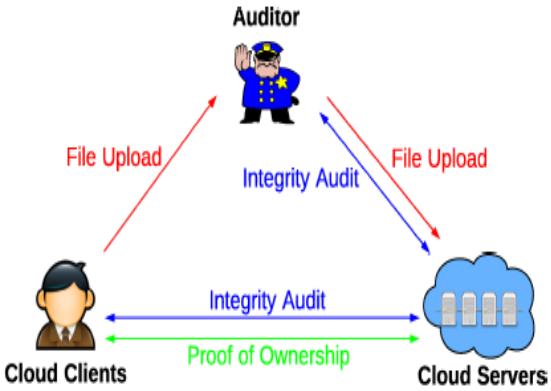

Figure 1: Architecture

SecCloud and SecCloud+ systems supports file level deduplication by including the following three protocols respectively highlighted by red, blue and green in figure:

1. File Uploading Protocol: This protocol aims at allowing clients to upload files via the auditor. Specifically, the file uploading protocol includes three phases:

Phase 1 (cloud client → cloud server): Before uploading a file, client performs the duplicate check with the cloud server to confirm if such a file is stored in cloud storage or not. If there is a duplicate, another protocol called Proof of Ownership will be run between the client and the cloud storage server. Otherwise, the following protocols (including phase 2 and phase 3) are run between these two entities

Phase 2 (cloud client → auditor): client uploads files to the auditor, and receives a receipt from auditor.

Phase 3 (auditor → cloud server): auditor helps generate a set of tags for the uploading file, and send them along with this file to cloud server.

2. Integrity Auditing Protocol: This protocol performs integrity verification and can be initialized by any entity except the cloud server. In this protocol, the cloud server plays the role of prover, while the auditor or client works as the verifier. This protocol includes two phases:

Phase 1 (cloud client/auditor → cloud server): verifier (i.e., client or auditor) generates a set of challenges and sends them to the prover (i.e., cloud server).

Phase 2 (cloud server → cloud client/auditor): 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) based on the stored files and file tags. At the end of this protocol, verifier outputs true if the integrity verification is passed.

3. Proof of Ownership Protocol: This protocol is initialized at the cloud server for verifying that the client exactly owns a claimed file. This protocol is typically triggered along with file uploading protocol to prevent the leakage of side channel information. In PoW the cloud server works as verifier, while the client plays the role of prover. This protocol also includes two phases:

Phase 1 (cloud server → client): Cloud server generates a set of challenges and sends them to the client.

Phase 2 (client → cloud server): The client responds with the proof for file ownership, and cloud server finally verifies the validity of proof.

V. FUNCTIONAL REQUIREMENTS

- Your Registration for a new Cloud Client is provided.
- Clients are able to upload/download files in SecCloud and SecCloud + systems via Auditor.
- Instead of storing a duplicate file on server. Proof of Ownership protocol will run to verify owners identity by requesting for privilege key.
- If privilege key is verified, reference ID of existing file is given to client. Thereby achieving data deduplication.
- Auditor sees request from clients to upload files and, uploads them on server by generating tags.
- Client and Auditor are able to perform integrity auditing for any block of data stored on server.
- Client can view the files uploaded on server.
- Clients are able to upload encrypted files on server using SecCloud+ system.
- Clients can download encrypted files from server by decrypting files using private key in SecCloud+ system.
- Clients /Auditor are able to perform integrity auditing and data deduplicating on encrypted data.

VI. NON FUNCTIONAL REQUIREMENTS

- *Consistency*
  The system must be consistent. It should not have any erratic behaviour during operation.
- *Error Prevention and Correction*
  The system is capable of handling large number of clients. If any error occurs the system should show the statement containing error. It should take preventive action to resolve it.
- *Maintenance*
  Monitor the maintenance of the project. Without Maintenance, there is a risk that project staff will remain responsible for maintenance or that the product will not be maintained at all Project deliverables that require maintenance after implementation should be addressed
- *Extensibility*
  Defining the desired outcomes or acceptance criteria at the beginning of the project also clarifies the project's scope. Using performance measures ascertains whether the project did indeed succeed, and provides a starting point for developing future lessons learned.

VII. EXTERNAL INTERFACE REQUIREMENTS

A. *User Interface:*

The application that will be developing will have a user friendly and menu based interface. Following screens will be provided for Customers:

- A login screen for Cloud Clients, Auditors and Cloud Servers to enter the username and password, so that the authorized entity can access without any problems.
- For all services, there exist different page.
- Security has been included for all types of users.

B. *Software Requirements:*

- Operating System: Windows XP or Higher Version.
- User Interface : Html/CSS
- Client Side Scripting : Java Script
- Programming Language : Java
- Web Application: JDBC, Servlet and JSP
- IDE : EditPlus
- Database: MySql
- Server Deployment : Tomcat 7.0

C. *Hardware Requirements:*

- Processor : Pentium IV or Higher processor
- RAM : 512 MB or More
- Hard disk : Minimum 40 GB

VIII. CONCLUSION

In this Paper, data integrity and deduplication in cloud is achieved by proposing SecCloud and SecCloud+ models. SecCloud proposes an auditing entity with maintenance of a MapReduce cloud helping clients generate data tags before uploading as well as integrity auditing of data having been stored in cloud. In addition, SecCoud enables secure deduplication by introducing a Proof of Ownership protocol and preventing the leakage of side channel information in data deduplication. Compared with previous work, during the file uploading and auditing phases computation by user in SecCloud is greatly reduced. SecCloud+ is proposed by the fact that customers always want to encrypt their data before uploading, and allows for integrity auditing and secure deduplication directly on encrypted data. This system can be implemented on real time cloud

ACKNOWLEDGEMENT

It would not have been possible without the kind support and help of many individuals who have directly or indirectly helped me in completing the paper. I would like to extend my sincere thanks to all of them.

I thank Almighty God for everything.

I express my immense gratitude to my guide Dr. Mohammed Abdul Waheed for his incredible support.

I express my deep gratitude to Dr. Shubhangi D.C, HOD and all the faculty of department for their cooperation.

My heartfelt gratitude towards my family and friends for their kind cooperation and encouragement in helping to complete my paper.

REFERENCES

[1] M.Armbrust A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski,G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "A view of cloud computing," Communication of the ACM

[2] J. Yuan and S. Yu, "Secure and constant cost public cloud storage auditing with deduplication," in IEEE Conference on Communications and Network Security (CNS), 2013

[3] S. Halevi, D. Harnik, B. Pinkas, and A. Shulman-Peleg, "Proofs of ownership in remote storage systems," in Proceedings of the 18th ACM Conference on Computer and Communications Security

[4] S. Keelveedhi, M. Bellare, and T. Ristenpart, "Dupless: Server- aided encryption for deduplicated storage," in Proceedings of the 22Nd USENIX Conference on Security, ser. SEC'13. Washington, D.C.: USENIX Association, 2013, pp. 179-194. [Online].

[5] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, "Provable data possession at untrusted stores," in Proceedings of the 14th ACM Conference on Computer and Communications Security, ser. CCS '07. New York, NY, USA: ACM, 2007, pp. 598- 609.

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

[7] G. Ateniese, R. Di Pietro, L. V. Mancini, and G. Tsudik, "Scalable and efficient provable data possession," in Proceedings of the 4th International Conference on Security and Privacy in Communication Netowrks, ser. SecureComm '08. New York, NY, USA: ACM, 2008, pp. 9:1-9:10.

[8] C. Erway, A. Kupcu, C. Papamanthou, and R. Tamassia, "Dynamic ¨ ¸¨ provable data possession," in Proceedings of the 16th ACM Conference on Computer and Communications Security, ser. CCS '09. New York, NY, USA: ACM, 2009, pp. 213-222

[9] H. Wang, "Proxy provable data possession in public clouds," IEEE Transactions on Services Computing, vol. 6, no. 4, pp. 551-559, 2013.

[10] Y. Zhu, H. Hu, G.-J. Ahn, and M. Yu, "Cooperative provable data Possession for integrity verification in multicloud storage," IEEE Trans- actions on Parallel and Distributed Systems, vol. 23, no. 12, pp. 2231- 2244, 2012

[11] Proceedings of the 14th International Conference on the Theory and Application of Cryptology and Information Security: Advances in Cryp- tology, ser. ASIACRYPT '08. Springer Berlin Heidelberg, 2008, pp. 90-107.

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

[13] J. Xu and E.-C. Chang, "Towards efficient proofs of retrievability," in Proceedings of the 7th ACM Symposium on Information, Computer and Communications Security, ser. ASIACCS '12. New York, NY, USA: ACM, 2012, pp. 79-80.

[14] R. Di Pietro and A. Sorniotti, "Boosting efficiency and security in proof of ownership for deduplication," in Proceedings of the 7th ACM Symposium on Information, Computer and Communications Security, ser. ASIACCS '12. New York, NY, USA: ACM, 2012, pp. 81-82.

[15] J. Douceur, A. Adya, W. Bolosky, P. Simon, and M. Theimer, "Reclaim- ing space from duplicate files in a serverless distributed file system," in 22nd International Conference on Distributed Computing Systems, 2002, pp. 617-624.

[16] J. Li, X. Chen, M. Li, J. Li, P. Lee, and W. Lou, "Secure deduplication with efficient and reliable convergent key management," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 6, pp. 1615-1625, June 2014.

[17] R. Di Pietro and A. Sorniotti, "Boosting efficiency and security in proof of ownership for deduplication," in *Proceedings of the 7th ACM Symposium on Information, Computer and Communications Security*, ser. ASIACCS '12. New York, NY, USA: ACM, 2012, pp. 81-82.

[18] W. K. Ng, Y. Wen, and H. Zhu, "Private data deduplication protocols in cloud storage," in *Proceedings of the 27th Annual ACM Symposium on Applied Computing, ser. SAC '12. New York, NY, USA: ACM,*

[19] J. Douceur, A. Adya, W. Bolosky, P. Simon, and M. Theimer, Reclaiming space from duplicate files in a serverless distributed file system," in *22nd International Conference on Distributed Computing Systems*, 2002, pp. 617-624.

[20] M. Bellare, S. Keelveedhi, and T. Ristenpart, "Message-locked encryption and secure deduplication," in *Advances in Cryptology - EURO- CRYPT 2013*, ser. Lecture Notes in Computer Science, T. Johansson and P. Nguyen, Eds. Springer Berlin Heidelberg, 2013,

