ISSN: 2278 – 909X
International Journal of Advanced Research in Electronics and Communication Engineering (IJARECE)
Volume 5, Issue 5, May 2016

SecCloud and SecCloud+ in Secure Auditing and Deduplicating data in Cloud

Ramanjaneya Reddy D (MTech, CSE , CMRIT), **Assoc Prof Swathi Y**(HOD, Dept of CSE, CMIRT)

Abstract: In recent trend, the huge data is being outsourced to cloud. Which decreases the burden of sparing efforts on heavy data protection and administration. outsourced cloud storing is not fully dependable, it raises security issues on how to realize data deduplication in cloud while obtaining integrity auditing. In this paper, we study the problematic integrity auditing and secure deduplication in the cloud data. Exactly, aiming at getting both data integrity and deduplication in cloud, we present two secure methods, namely SecCloud and SecCloud+. SecCloud introduces an auditing entity with a protection of a MapReduce cloud, which benefits users create data tags before uploading as well as audit the integrity of data having been saved in cloud. Related with previous work, the computation by user in SecCloud is greatly reduced throughout the file uploading and auditing stages. SecCloud+ is designed motivated by the fact that customers always want to encode their data before uploading, and allows integrity auditing and secure deduplication on encrypted data.

Keywords: Cloud Storage, Data deduplicating, Secure auditing.

I. INTRODUCTION

Cloud storage is a standard of networked enterprise storing where data is stored in virtualized pools of storage which are usually hosted by third parties. Cloud storage provides customers with welfares, ranging from price saving and simplified convenience, to mobility chances and scalable service. These great properties attract more and more customers to use and storing their personal data to the cloud storage: agreeing to the analysis report, the volume of data in cloud is likely to reach 40 trillion gigabytes in 2020. Even though cloud storage system has been extensively adopted, it fails to adapt some main emerging needs such as the abilities of auditing integrity of cloud files by cloud clients and detecting repeated files by cloud servers. We illustrate both problems below. The first problem is integrity auditing. The cloud server is able to release clients from the heavy load of storage management and maintenance. The main distinction of cloud storage from old method in-house storage is that the data is transported via Internet and deposited in an uncertain domain, not in the control of the clients at all, which inevitably raises clients great worries on the integrity of their data. These concerns originate since the fact that the cloud storage is susceptible to security fears from both outside and inside of the cloud [1], and the unrestrained cloud servers may passively hide certain data loss actions from the clients to maintain their name. What is more serious is that for saving money and space, the cloud servers force even actively and deliberately reject rarely retrieved data files belonging to an normal client. Considering the huge size of the outsourced data files and clients' controlled resource capabilities, the first problematic is generalized as how can the client efficiently achieve periodical integrity verifications even without the local copy of data files.

The second problematic is secure deduplication. The rapid adoption of cloud services is attended by increasing volumes of data deposited at remote cloud servers. Among these remote stored files, most of them are repeated: according to a last review by EMC [2], 75% of current digital data is duplicated copies. This fact raises a technology namely deduplication, in which the cloud servers will be like to deduplicate by keeping only a single copy for each file and make a link to the file for each client who owns or asks to store the similar file. Unfortunately, this action of deduplication would lead to a number of risks possibly affecting the storage system [3][2], for example, a server saying a client that it (i.e., the client) does not necessity to send the file reveals that some other client has the similar file, which might be delicate sometimes. These attacks originate from the reason that the proof that the client holds a given file (or block of data) is only based on a static, short value [3] (in most cases the hash of the file). Thus, the second problematic is simplified as how can the cloud servers efficiently confirm that the client owns the file which is uploaded before creating a link to this file for him/her.
In this paper, aiming at reaching data integrity and deduplication in cloud, we propose two secure methods namely SecCloud and SecCloud+. SecCloud presents an auditing entity with a preservation of a MapReduce cloud, which supports clients creating data tags before uploading as well as audit the integrity of data is been saved in cloud. This design shows the problem of previous work that the computational burden at user or auditor is too huge for tag

Manuscript received May, 2016.
***Ramanjaneya Reddy D**, Computer Science,CMR Institute Of Technology, Banglore,India,7760320004.*
***Prof. Swathi Y**, Computer Science, CMR Institute Of Technology, Banglore, India, +919900759571.*

formation. For completeness of fine-grained, the function of auditing designed in SecCoud is maintained on both block level and sector level. In addition, SecCoud also permits secure deduplication. Notice that the "security" considered in SecCoud is the prevention of leak of cross channel information. In order to stop the leakage of such side channel information, we follow the old method of [3][2] and design a proof of ownership protocol in between clients and cloud servers, which permits clients to prove to cloud servers that they accurately own the target data. Motivated by the fact that customers every time want to encode their data before uploading, for purposes ranging from personal privacy to business policy, we present a key server into SecCloud as with [4] and offer the SecCloud+ schema. Besides supporting integrity auditing and secure deduplication, SecCloud+ permits the promise of file confidentiality. Specifically, thanks to the property of deterministic encryption in convergent encryption, we present a method of straight auditing integrity on encrypted data. The challenge of deduplication on encrypted is the avoidance of dictionary attack [4]. As with [4], we make a change on convergent encryption such that the convergent key of file is created and coordinated by a secret "seed", such that any adversary could not directly originate the convergent key from the content of file and the dictionary attack is prevented.

II. Related Work

Our work is being related to both integrity auditing and secure deduplication, we review the works in both areas in the following the subsets, respectively.

A. Integrity Auditing

The description of provable data possession (PDP) was developed by Ateniese et al. [5][6] for promising that the cloud servers possess the focus files without recovering or downloading the entire data. Basically, PDP is a probabilistic proof protocol by sample a random set of blocks and requesting the servers to prove that they precisely possess these blocks, and the verifier only upholding a small amount of metadata is able to achieve the integrity checking. After Ateniese et al.'s proposal [5], some works worried on how to realize PDP on vigorous scenario: Ateniese et al. [7] projected a dynamic PDP plan but without insertion operation; Erway et al. [8] improved Ateniese et al.'s work [7] and maintained insertion by presenting authenticated flip table; A related work has also been given in [9]. Nevertheless, these proposals [5][7][8][9] suffer from the computational overhead for tag generation at the client. To solve this issue, Wang et al. [10] obtainable proxy PDP in public clouds. Zhu et al. [11] presented the helpful PDP in multi-cloud storage.

Another line of work associate integrity auditing is proof of retrievability (POR) [12]. Related with PDP, POR not merely assures the cloud servers hold the target files, but also guarantees their complete recovery. [12] In clients apply erasure codes and make authenticators for every block for verifiability and retrievability. In order to get effective data dynamics, Wang et al. [13] improved the POR model by manipulating the classic Merkle hash tree creation for block tag authentication. Xu and Chang [14] presented to progress the POR schema in [12] with polynomial commitment for dropping communication cost. Stefanov et al. [15] proposed a POR protocol over authenticated file system subject to regular changes. Azraoui et al. [16] merged the privacy-preserving each word search algorithm with the addition in data segments of randomly created short bit sequences, and developed a new POR protocol. Li et al. [17] measured a new cloud storage architecture with two independent cloud servers for integrity auditing to decrease the computation burden at client side. Recently, Li et al. [18] used the key-disperse paradigm to solve the issue of a significant number of convergent keys in convergent encryption.

B. Secure Dedublication

Deduplication is a method where the server stores only a one copy of each file, regardless of which clients asked to store that file, such that the storage disk space of cloud servers as well as network bandwidth are reduced. However, trivial client side deduplication leads to the leak of side channel information. For example, a server signifying a client that it is not required to send the file will disclose that some other client has the exact same file, which could be delicate information in some case.

In order to limit the leakage of side channel information, Halevi et al. [3] presenting the proof of ownership protocol which allows a client effctive show to a server that that the client exactly holds this file. Some proof of ownership of this protocols are based on the Merkle hash tree are proposed [3] to permit secure client-side deduplication. Sorniotti and Pietro [19] proposed an effective proof of ownership scheme by selecting the projection of a file onto few randomly selected are bit- positions as it is the file proof. Another line of work for the secure deduplication which focuses on the confidentiality of deduplicated data and reflects to make deduplication on encrypted data. [20] Ng et al. firstly introduced the private data deduplication as a accompaniment of public data deduplication protocols of [3] Halevi et al. Convergent encryption [21] is a assuring cryptographic primeval for certifying data privacy in deduplication. Bellare et al. [22] formalized this primitive as message-locked encryption, and discovered its application in space-efficient secure outsourced storage. Abadi et al. [23] further strengthened Bellare et al's security definitions [22] by considering plaintext distributions that may differ on the public parameters of the schemas. About the practical deployment of convergent encryption for securing deduplication, Keelveedhi et al. [4] designed the DupLESS model in which clients encrypt under file-based keys derived from a key server via an oblivious pseudorandom function protocol.

As specified before, all the works illustrated above believes either deduplication or integrity auditing, where in this paper, we attempt to solve two problems simultaneously. In addition, it is worthwhile noticing that our work is also distinguished with [2] which audits cloud data along with deduplication, because we also consider to 1) audit and deduplicate encrypted data in the proposed protocols, 2) outsource the computation of tag generation.

III. PRILIMINARY

A. Bilinear Map and Computational Assumption

Definition 1: (Bilinear Map): Let G and GT be 2 cyclic multiplicative groups of large prime order p. A bilinear pairing is a map e : G × G → GT with the subsequent properties:

• Bilinear: e(g1 a, g2 b) = e(g1, g2)ab for all g1, g2 ∈ R G and a, b ∈ R Zp;

• Non-degenerate: There occurs g1, g2 ∈ G such that(g1, g2)≠ 1;

• Computable: There is effective algorithm to calculate e(g1, g2) for all g1, g2 ∈ R G.

The examples of such groups can be found in great singular elliptic curves or hyper elliptic curves throughout finite fields, and the bilinear pairings can be originated from the Tate pairings or Weil. For more details, see [24].

We then describe the Computational Diffie-Hellman difficulty, the rigidity it will be the basis of the security of our proposed schemes.

Definition2: (CDH Problem): The Computational DiffieHellman problematic is that, given g, gx, gy ∈ G1 for unknown x, y ∈ Z∗ p, to calculate gxy.

B. Convergent Encryption

Convergent encryption [22][23][21] delivers data confidentiality in deduplication. A user (or data owner) derives a convergent key from the data content and encrypts the data reproduction with the convergent key. In addition, the user will derives a tag for the data copy, such that the tag will be used to identify duplicates. Here, we accept that the tag correctness property [22] holds, i.e., if two data copies are the same, then their tags are the same. Formally, a convergent encryption scheme can be defined with four primitive functions:

• KeyGen(F) : The key creation algorithm receives a file content F as output and inputs the convergent key ckF of F;

• Encrypt(ckF, F) : The encryption algorithm receives convergent key i.e ckF and file content F as input and outputs the ciphertext ctF;

• Decrypt(ckF, ctF) : The decryption algorithm receives the ciphertext ctF and convergent key ckF as input and outputs the plain file F;

• TagGen(F) : The tag generation algorithm recives a file content F as input and outputs the tag tagF of F. Signs that in this paper, we also permit TagGen(·) to generate the (same) tag from the equivalent ciphertext as with [22][18].

IV. SYSTEM MODEL

Aiming at permitting for auditable and deduplicated storage, we suggest the SecCloud system. In the SecCloud system, we have 3 entities:

A. *Integrity Clients*

Cloud Clients have huge data files to be stored and depend on the cloud for data computation and maintenance. They can be either specific consumers or business organizations.

B. *Cloud Servers*

Cloud Servers virtualize the resources according to the requirements of the clients and project them as storage pools. Normally, the cloud clients might buy or rent storage capacity from cloud servers, and stores their individual data in these bought or lease spaces for upcoming utilization.

C. *Auditor*

Auditor which benefits clients upload and audit their outsourced data supports a MapReduce cloud and acts like a licensed authority. This assumption believes that the auditor is related with a pair of private and public keys. Its public key is made available to the other entities in the system.

FIGURE 1: SecCloud Architecture

The SecCloud scheme supporting file-level deduplication includes the following 3 protocols respectively highlighted by red, blue and green in Fig.[25]

1) FILE UPLOADING PROTOCOL:

This protocol aims at permitting clients to upload files via the auditor. Precisely, the file uploading protocol includes 3 phases:

I)PHASE 1 (CLOUD CLIENT → CLOUD SERVER): User takes the identical check with the cloud server to approve if such a file is saved in cloud storage or not before uploading a file. If there is a identical file, another procedure called Proof of Ownership will be taken between the user and the cloud storage server. Then, the following protocols (including phase 3 and phase 2) are run between these two units.

II) PHASE 2 (CLOUD CLIENT → AUDITOR): Client uploads files to the auditor, and receives a receipt from auditor.

III)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: It is a collaborative protocol for reliability verification and allowed to be initialized by any unit except the cloud server. In this procedure, the cloud server. plays the role of approver, while the assessor or user works as the authenticator. This protocol includes two phases:

I) PHASE 1 (CLOUD CLIENT/AUDITOR → CLOUD SERVER): Authenticator (i.e., user or assessor) generates a set of challenges and sends them to the approver (i.e., cloud server).

II) PHASE 2 (CLOUD SERVER → CLOUD CLIENT/AUDITOR): Based on the saved files and file tags/names, approver (i.e., cloud server) tries to authenticate that it exactly owns the target file by transmitting the proof back to Authenticator (i.e., cloud user or assessor). At the end of this procedure, approver outputs true if the reliability verification is a success.

3) PROOF OF OWNERSHIP PROTOCOL:

It is a collaborative procedure set at the cloud server for verifying that the client exactly owns a claimed file. This procedure is usually triggered along with file upload procedure to avoid the leakage of side channel data. On the contrast to reliability reviewing procedure, in PoW the cloud server works as authenticator, while the client plays the role of approver. This procedure also includes two phases

I) PHASE 1 (CLOUD SERVER → CLIENT): Cloud server produces a set of tasks and transmits them to the user.

II)PHASE 2 (CLIENT → CLOUD SERVER): The user responds with the evidence for file ownership, and cloud server finally confirms the validity of proof. Our initial purposes are as follows.

I) INTEGRITY AUDITING:

The 1st design goal of this work is to deliver the ability of validation and authentication of the remotely saved information. The reliability authentication further involves 2 features which are public authentication and stateless validation.

II) SECURE DEDUPLICATION:

The 2nd design aim of this process is secure deduplication. In other meaning, it obliges that the cloud server is able to limit the storage space by keeping only a single copy of the uploaded file. Observe that, considering to the secure deduplication, our objective is differentiated from prior work [3] in that we scheme a process for accepting both deduplication over files and tags.

III) COST-EFFECTIVE:

The computational process for specifying reliably auditing and secure deduplication should not display a huge overhead cost to already available cloud storage, neither should they change the way either uploading or downloading operation.

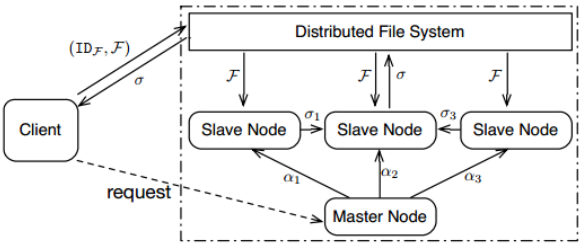

FIGURE 2: FILE UPLOADING PROTOCOL

1. File Uploading Protocol: In the first phase shown in Fig. 2, the user runs the deduplication process by sending hash value of the existing file Hash(F) to the cloud server. If there is a replica, the cloud user achieves Proof of Ownership protocol with the cloud server.

In the 2nd phase the cloud client stores a file F as well as its identity IDF to the distributed file system in MapReduce auditing cloud.

V. CONCLUSION

Directing at checking both data integrity and deduplication in cloud, we present SecCloud and SecCloud+. SecCloud defines an assessing entity with preservation of a MapReduce cloud, which helps users create information tags before saving as well as audit the integrity of information having been saved in cloud. In addition, SecCoud permits secure deduplication through providing a Proof of Ownership procedure and preventing the outflow of side channel data in data deduplication. Paralleled with previous work, the

calculation by client in SecCloud is greatly reduced throughout the file uploading and auditing process. SecCloud+ is an sophisticated superior creation motivated by the fact that consumers always want to encrypt their information before uploading, and authenticating or validating for integrity auditing and secure deduplication directly on encrypted information.

ACKNOWLEDGMENT

I dedicate all our research to our esteemed guide, Prof. Swathi Y (H.O.D. Computer Engineering Dept.), whose concern, attentiveness and guidance helped us to complete the research successfully. This experience will always help us to do our work perfectly and professionally with integrity.

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, vol. 53, no. 4, pp. 50–58, 2010.

[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, pp. 145–153.

[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. ACM, 2011, pp. 491–500.

[4] S. Keelveedhi, M. Bellare, and T. Ristenpart, "Dupless: Serveraided 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]. Available: https://www.usenix.org/conference/usenixsecurity13/technicalsessions/presentation/bellare

[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. Kupc¸ ¨ u, 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] F. Sebe, J. Domingo-Ferrer, A. Martinez-Balleste, Y. Deswarte, and ´ J.-J. Quisquater, "Efficient remote data possession checking in critical information infrastructures," IEEE Trans. on Knowl. and Data Eng., vol. 20, no. 8, pp. 1034–1038, 2008.

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

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

[12] H. Shacham and B. Waters, "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 '08. Springer Berlin Heidelberg, 2008, pp. 90–107.

[13] 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.

[14] 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.

[15] E. Stefanov, M. van Dijk, A. Juels, and A. Oprea, "Iris: A scalable cloud file system with efficient integrity checks," in Proceedings of the 28th Annual Computer Security Applications Conference, ser. ACSAC '12. New York, NY, USA: ACM, 2012, pp. 229–238.

[16] M. Azraoui, K. Elkhiyaoui, R. Molva, and M. Onen, "Stealthguard: ¨ Proofs of retrievability with hidden watchdogs," in Computer Security - ESORICS 2014, ser. Lecture Notes in Computer Science, M. Kutyłowski and J. Vaidya, Eds., vol. 8712. Springer International Publishing, 2014, pp. 239–256.

[17] J. Li, X. Tan, X. Chen, and D. Wong, "An efficient proof of retrievability with public auditing in cloud computing," in 5th International Conference on Intelligent Networking and Collaborative Systems (INCoS), 2013, pp. 93–98.

[18] 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.

[19] 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.

[20] 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, 2012, pp. 441–446.

[21] 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.

[22] M. Bellare, S. Keelveedhi, and T. Ristenpart, "Message-locked encryption and secure deduplication," in Advances in Cryptology – EUROCRYPT 2013, ser. Lecture Notes in Computer Science, T. Johansson and P. Nguyen, Eds. Springer Berlin Heidelberg, 2013, vol. 7881, pp. 296–312.

[23] M. Abadi, D. Boneh, I. Mironov, A. Raghunathan, and G. Segev, "Message-locked encryption for lock-dependent messages," in Advances in Cryptology – CRYPTO 2013, ser. Lecture Notes in Computer Science, R. Canetti and J. Garay, Eds. Springer Berlin Heidelberg, 2013, vol. 8042, pp. 374–391.