ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

An Approved Mechanism to Evaluate the Redundant Data in Cloud

D.Saika
M.Tech Student
Vignanabharathi Institute of Technology And Sciences,
Ghatkesar.

V.Sridhar Reddy
Associate Profesor
Vignanabharathi Institute of Technology And Sciences,
Ghatkesar.

Abstract

Outsourcing data to cloud service for storage becomes an important trend, which benefits in sparing efforts on heavy data maintenance and management. The outsourced cloud storage is not fully trustworthy; it raises security concerns on how to realize data deduplication in cloud while getting integrity auditing. In this paper, we study the problem of integrity auditing and secure deduplication on cloud data. Specifically, aiming at getting both data integrity and deduplication in cloud, we present two secure systems, namely SecCloud and SecCloud+. SecCloud introduces an auditing entity with a maintenance of a MapReduce cloud, which helps clients create data tags before uploading as well as audit the integrity of data having been saved 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.

Key Words: ***Cloud Storage, Data deduplicating, Secure auditing.***

INTRODUCTION

Cloud storage is a model of networked enterprise storage where data is stored in virtualized pools of storage which are generally hosted by third parties. Cloud storage provides customers with benefits, ranging from cost saving and simplified convenience, to mobility opportunities and scalable service. These great properties attract more and more customers to use and storage their personal data to the cloud storage: according to the analysis report, the volume of data in cloud is expected to achieve 40 trillion gigabytes in 2020. Even though cloud storage system has been widely adopted, it fails to accommodate some main emerging needs such as the abilities of auditing integrity of cloud files by cloud clients and detecting duplicated files by cloud servers. We illustrate both problems below. The first problem is integrity auditing. The cloud server is able to relieve clients from the heavy burden of storage management and maintenance. The main difference of cloud storage from traditional in-house storage is that the data is transferred via Internet and stored in an uncertain domain, not under control of the clients at all, which inevitably raises clients great concerns on the integrity of their data. These concerns originate from the fact that the cloud storage is susceptible to security threats from both outside and inside of the cloud [1], and the uncontrolled cloud servers may passively hide some data loss incidents from the clients to maintain their reputation. What is more serious is that for saving money and space, the cloud servers might even actively and deliberately discard rarely accessed data files belonging to an ordinary client. Considering the large size of the outsourced data files and the clients' constrained resource capabilities, the first problem is

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

generalized as how can the client efficiently perform periodical integrity verifications even without the local copy of data files.

The second problem is secure deduplication. The rapid adoption of cloud services is accompanied by increasing volumes of data stored at remote cloud servers. Among these remote stored files, most of them are duplicated: according to a last survey by EMC [2], 75% of recent digital data is duplicated copies. This fact raises a technology namely 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 the same file. Unfortunately, this action of deduplication would lead to a number of threats potentially affecting the storage system [3][2], for example, a server telling a client that it (i.e., the client) does not need to send the file reveals that some other client has the same file, which could be sensitive sometimes. These attacks originate from the reason that the proof that the client owns a given file (or block of data) is solely based on a static, short value (in most cases the hash of the file) [3]. Thus, the second problem is generalized as how can the cloud servers efficiently confirm that the client owns the uploaded file before creating a link to this file for him/her.

In this paper, aiming at getting data integrity and deduplication in cloud, we present two secure systems namely SecCloud and SecCloud+. SecCloud introduces an auditing entity with a maintenance of a MapReduce cloud, which helps clients create data tags before uploading as well as audit the integrity of data having been saved in cloud. This design shows the issue of previous work that the computational load at user or auditor is too large for tag creation. For completeness of fine-grained, the functionality of auditing designed in SecCoud is supported on both block level and sector level. In addition, SecCoud also enables secure deduplication. Notice that the "security" considered in SecCoud is the prevention of leakage of side channel information. In order to avoid the leakage of such side channel information, we follow the tradition of [3][2] and design a proof of ownership protocol between clients and cloud servers, which permits clients to prove to cloud servers that they exactly own the target data. Motivated by the fact that customers always want to encrypt their data before uploading, for reasons ranging from personal privacy to corporate policy, we present a key server into SecCloud as with [4] and propose the SecCloud+ schema. Besides supporting integrity auditing and secure deduplication, SecCloud+ enables the guarantee of file confidentiality. Specifically, thanks to the property of deterministic encryption in convergent encryption, we presente a technique of directly auditing integrity on encrypted data. The challenge of deduplication on encrypted is the prevention of dictionary attack [4]. As with [4], we make a modification on convergent encryption such that the convergent key of file is created and controlled by a secret "seed", such that any adversary could not directly derive the convergent key from the content of file and the dictionary attack is prevented.

RELATED WORK

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

Intigrity Auditing

The definition of provable data possession (PDP) was developed by Ateniese et al. [5][6] for assuring that the cloud servers possess the target files without retrieving or downloading the whole data. Essentially, PDP is a probabilistic proof protocol by sampling a random set of blocks and asking the servers to prove that they exactly possess these blocks, and the verifier only maintaining a small amount of metadata is able to perform the integrity checking. After Ateniese et al.'s proposal [5], several works concerned on how to realize PDP on dynamic scenario: Ateniese et al. [7] proposed a dynamic PDP schema but without insertion operation; Erway et al. [8] improved Ateniese et al.'s work [7] and supported insertion by introducing authenticated flip table; A similar work has also been contributed in [9]. Nevertheless, these proposals

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

[5][7][8][9] suffer from the computational overhead for tag creation at the client. To fix this issue, Wang et al. [10] presented proxy PDP in public clouds. Zhu et al. [11] presented the cooperative PDP in multi-cloud storage.

Another line of work supporting integrity auditing is proof of retrievability (POR) [12]. Compared with PDP, POR not merely assures the cloud servers possess the target files, but also guarantees their full recovery. In [12], clients apply erasure codes and create authenticators for each block for verifiability and retrievability. In order to get efficient data dynamics, Wang et al. [13] improved the POR model by manipulating the classic Merkle hash tree construction for block tag authentication. Xu and Chang [14] presented to improve the POR schema in [12] with polynomial commitment for reducing communication cost. Stefanov et al. [15] proposed a POR protocol over authenticated file system subject to frequent changes. Azraoui et al. [16] combined the privacy-preserving word search algorithm with the insertion in data segments of randomly created short bit sequences, and developed a new POR protocol. Li et al. [17] considered a new cloud storage architecture with two independent cloud servers for integrity auditing to reduce the computation load at client side. Recently, Li et al. [18] used the key-disperse paradigm to fix the issue of a significant number of convergent keys in convergent encryption.

Secure Deduplication

Deduplication is a method where the server saves only a single copy of each file, regardless of which 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, Halevi et al. [3] introduced the proof of ownership protocol which lets a client efficiently prove to a server that that the client exactly holds this file. Several proof of ownership protocols based on the Merkle hash tree are proposed [3] to enable secure client-side deduplication. 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. Another line of work for secure deduplication focuses on the confidentiality of deduplicated data and considers to make deduplication on encrypted data. Ng et al. [20] firstly introduced the private data deduplication as a complement of public data deduplication protocols of Halevi et al. [3]. Convergent encryption [21] is a promising cryptographic primitive for ensuring data privacy in deduplication. Bellare et al. [22] formalized this primitive as message-locked encryption, and explored 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 depend on the public parameters of the schemas. Regarding the practical implementation of convergent encryption for securing deduplication, Keelveedhi et al. [4] designed the DupLESS system in which clients encrypt under file-based keys derived from a key server via an oblivious pseudorandom function protocol.

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

Existing System

Cloud storage is a model of networked enterprise storage where data is stored in virtualized pools of storage which are generally hosted by third parties. Cloud storage provides customers with benefits, ranging from cost saving and simplified convenience,

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

to mobility opportunities and scalable service. These great features attract more and more customers to utilize and storage their personal data to the cloud storage: according to the analysis report, the volume of data in cloud is expected to achieve 40 trillion gigabytes in 2020.

Disadvantages

Even though cloud storage system has been widely 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 illustrate both problems below.

- The first problem is integrity auditing
  How can the client efficiently perform periodical integrity verifications even without the local copy of data files. The second problem

- The second problem is secure deduplication
  How can the cloud servers efficiently confirm that the client (with a certain degree assurance) owns the uploaded file (or block) before creating a link to this file (or block) for him/her.

Proposed System

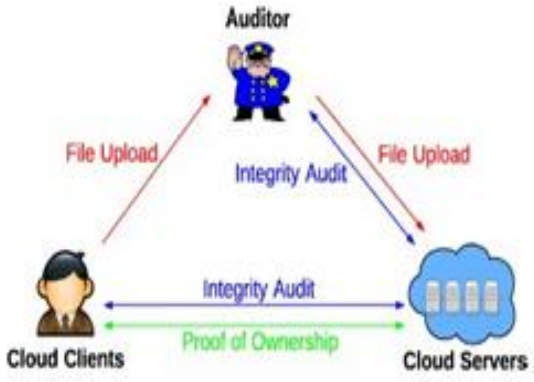
SecCloud introduces an auditing entity with maintenance of a MapReduce cloud, which helps clients generate data tags before uploading as well as audit the integrity of data having been stored in cloud. This design fixes the issue of previous work that the computational load at user or auditor is too huge for tag generation. For completeness of fine-grained, the functionality of auditing designed in SecCoud is supported on both block level and sector level. In addition, SecCoud also enables secure deduplication. Notice that the "security" considered in SecCoud is the prevention of leakage of side channel information. In order to prevent the leakage of such side channel information, the tradition of and design a proof of ownership protocol between clients and cloud servers, which allows clients to prove to cloud servers that they exactly own the target data. Motivated by the fact that customers always want to encrypt their data before uploading, for reasons ranging from personal SecCloud as with [4] and propose the SecCloud+ schema. Besides supporting integrity auditing and secure deduplication, SecCloud+ enables the guarantee of file confidentiality. Specifically, thanks to the property of deterministic encryption in convergent encryption, we propose a method of directly auditing integrity on encrypted data. The challenge of deduplication on encrypted is the prevention of dictionary attack. As with, we make a modification on convergent encryption such that the convergent key of file is generated and controlled by a secret "seed", such that any adversary could not directly derive the convergent key from the content of file and the dictionary attack is prevented.

Advantages

- The computation by user in SecCloud is greatly reduced during the file uploading and auditing phases.

- SecCloud+ is an advanced construction motivated 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.

System Architecture



ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

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

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

I)Phase 1 (cloud client → cloud server): Client takes the duplicate check with the cloud server to confirm if such a file is stored in cloud storage or not before uploading a file. 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.

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 an interactive protocol for integrity verification and allowed to 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:

I)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).

II)Phase 2 (cloud server → cloud client/auditor): 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). At the end of this protocol, verifier outputs true if the integrity verification is passed.

3)Proof of Ownership Protocol:
It is an interactive protocol 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. On the contrast to integrity auditing protocol, in PoW the cloud server works as verifier, while the client plays the role of prover. This protocol also includes two phases

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

II)Phase 2 (client → cloud server): The client responds with the proof for file ownership, and cloud server finally verifies the validity of proof. Our main objectives are as follows.

i)Integrity Auditing:
The first design goal of this work is to provide the capability of verifying correctness of the remotely stored data. The integrity verification further requires two features those are public verification and stateless verification.

ii)Secure Deduplication:
The second design goal of this work is secure deduplication. In other words, it requires that the cloud server is able to decrease the storage space by keeping only one copy of the same file. Notice that, regarding to secure deduplication, our objective is distinguished from previous work [3] in that we propose a method for allowing both deduplication over files and tags.

iii)Cost-Effective:
The computational overhead for providing integrity auditing and secure deduplication should not show a major additional cost to traditional cloud storage, nor should they alter the way either uploading or downloading operation.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

CONCLUSIONS

Aiming at getting both data integrity and deduplication in cloud, we present SecCloud and SecCloud+. SecCloud proposes an auditing entity with maintenance of a MapReduce cloud, which helps clients create data tags before uploading as well as audit the integrity of data having been stored in cloud. In addition, SecCoud enables secure deduplication through ipresenting a Proof of Ownership protocol and avoiding the leakage of side channel information in data deduplication. Compared with previous work, the computation by user in SecCloud is greatly decreased during the file uploading and auditing phases. SecCloud+ is an advanced construction motivated 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.

References

[1] Jingwei Li, Jin Li, Dongqing Xie and Zhang Cai "SECURE AUDITING AND DEDUPLICATING DATA IN CLOUD", IEEE Transactions on Computers Volume: PP , Issue: 99, 26 January 2015.

[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.179194.[Online].Available:https://www.usenix.org/co nference/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. K¨upc¸ ¨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. Seb´e, 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

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

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. O¨ nen, "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.