IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 5, Issue 5, May 2016

Integrity Auditing and Secure Deduplicating the Data on Cloud Storage

Mr. Lohith Kumar K K[1], Mr. Ashwin Kumar M[2]

Student of M.Tech (4th Semester), Dept. of Computer Science and Engineering, MITE Moodabidri, India[1]

Senior Assistant Professor, Dept. of Computer Science and Engineering, MITE Moodabidri, India[2]

Abstract: As cloud computing is a kind of Internet-based computing technology develops in recent years that provides shared processing assets and information as well as data to computers and other different devices on demand, outsourcing data to cloud service for storage becomes an important trend, which advantages in saving endeavors on substantial data maintenance and administration. The outsourced cloud storage is not completely reliable; it raises some security worries on how to realize data deduplication in cloud while getting integrity auditing. In this work, we study the issue of integrity auditing and secure deduplication of data on cloud storage. Specifically, aiming at accomplishing both data integrity and deduplication in cloud storage, we introduce two secure systems, namely SecCloud and SecCloud+. SecCloud presents an auditing entity with a maintenance of a MapReduce cloud, which helps clients create data tags before uploading a file as well as audit the integrity of data having been put in cloud. Compared with past work, the computation by user in SecCloud is extraordinarily diminished amid the file uploading and auditing phases. SecCloud+ is designed persuaded 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 Storage, outsourcing, data auditing, secure deduplication, SecCloud+.

I. INTRODUCTION

As the Cloud computing is a general form for the delivery of hosted services over the Internet. Cloud storage is a model of networked enterprise storage where data is stored in virtualized pools; a third-party provider facilitated and conveys as well as delivers the cloud service over the Internet. Cloud storage gives customers with advantages, ranging from cost saving and simplified convenience, to mobility opportunities and adaptable service.

These incredible properties draw in more clients to utilize and storage their personal data to the cloud storage: as indicated by the report, the volume of data in cloud is relied upon to accomplish 40 trillion gigabytes in 2020. Despite the fact that cloud storage system has been widely adopted, it fails to oblige some principle emerging needs for example, the capacities of auditing integrity of cloud files by cloud customers and detecting duplicated files by cloud servers. Both issues are illustrated below. The first issue 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 conventional in-house storage is that the data is exchanged via Internet and stored in an uncertain domain, not under control of the clients at all, which inevitably raises customers 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. In order to save money and space, the cloud servers might even dynamically and intentionally discard seldom accessed data files belonging to an ordinary client. Allowing for the large size of the outsourced data files and the clients forced resource capabilities, the first problem is generalized as how can the client efficiently perform periodical integrity verifications even without the local copy of data files.

The second issue is secure deduplication. The fast selection of cloud services is accompanied by growing volumes of data stored at isolated cloud servers. Among these isolated stored files, most of them are duplicated: according to a previous survey by EMC, 75% of recent digital data have duplicated copies.

This fact raises a technology namely deduplication, in which the cloud servers would like to deduplicate by keeping only a one copy for each file and make a link to the file for every client who owns or asks to store the same file. unhappily, 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.

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319 5940
International Journal of Advanced Research in Computer and Communication Engineering
Vol. 5, Issue 5, May 2016

In this paper, going for getting data integrity and deduplication in cloud, we exhibit two secure systems namely SecCloud and SecCloud+. SecCloud introduce 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 saved in cloud. This design shows the problem of previous work that the computational load at user or auditor is too huge for tag creation. For fulfilment of fine-grained, the functionality of auditing considered in SecCoud is supported on both block level and sector level. In addition, SecCoud also enables secure deduplication. Notice that the "security" reflect on in SecCoud is the avoidance 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 allows clients to prove to cloud servers that they exactly own the target data. Motivated by the truth that customers for all time want to encrypt their data before uploading, for reason ranging from personal privacy to corporate policy, we present a key server into SecCloud as with [4] and propose the SecCloud+ schema. also supporting integrity auditing and secure deduplication, SecCloud+ enables the assurance of file confidentiality. Specifically, thanks to the property of deterministic encryption in convergent encryption, we present 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 alteration on concurrent encryption such that the concurrent key of file is created and controlled by a secret "seed", such that any foe could straightforwardly derive the convergent key from the content of file and the dictionary attack is avoided.

II. SECCLOUD

In this section, we illustrate our proposed SecCloud system. In particular, we start with giving the system model of SecCloud also presenting the outline objectives of SecCloud.

A. System Model

Aiming at allowing for auditable and deduplicated storage, we propose the SecCloud system. In the SecCloud system, system include three entities

1) The Cloud Clients: Cloud Clients contain large data files to be uploading and rely on the cloud for data maintenance and computation. They can be either individual consumers or commercial organizations.

2) The Cloud Server: Cloud Servers virtualizes the resources according to the requirements of clients and expose them as storage pools. Typically, the cloud clients may purchase or rent stockpiling limit from cloud servers, and store their individual data in these purchased or leased space for future use.

3) The Auditor: Auditor which helps clients to upload and audit their outsourced data maintains a MapReduce cloud and acts like a certificate authority. This assumption presumes that the auditor is associated with a pair of public and private keys. Its public key is known to the other entities in the system.

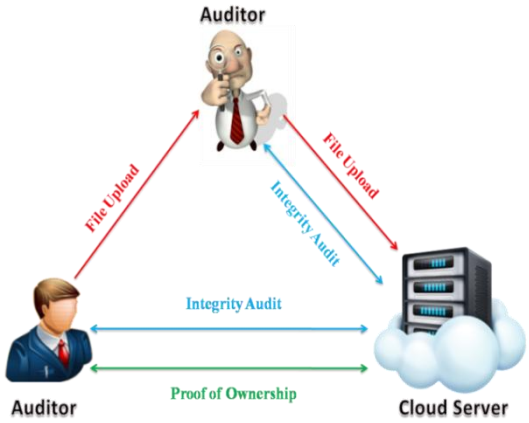

Fig. 1. Architecture of SecCloud.

The SecCloud system supporting file-level deduplication incorporates the following three protocols respectively highlighted by red, blue and green in Fig. 1.

1) File Uploading Protocol: In this protocol designed with the aims at allowing clients to upload files via the auditor. Specifically, the file uploading protocol contain three phases:

- Phase 1 (cloud client → cloud server): client performs the duplicate check with the cloud server to confirm if such a file is stored in cloud storage or not before uploading a file. On the off chance that there is a duplicate, another protocol called Proof of Ownership will be run between the client and the cloud storage server. Otherwise, the other protocols (including phase 2 and phase 3) will execute 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 make 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 permitted to be initialized by any entity except the cloud server. In this protocol, the cloud server assumes the part of prover,, while the auditor or client works as the verifier. This protocol again 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): based on the stored files and file tags, prover (i.e., cloud server) tries to confirm that it exactly owns the target file by sending the proof back to verifier (i.e., cloud client or auditor). At the ending of this protocol, verifier produces output as true if the integrity verification is passed.

3) Proof of Ownership Protocol: It is an intuitive protocol instated at the cloud server for confirming that the client precisely possesses claimed file. This protocol is commonly triggered along with file uploading protocol to avoid 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

- 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.

Our important objectives are outlined as follows.
1) Integrity Auditing: The first design goal of this work is to provide the ability of confirming correctness of the remotely stored data. The integrity verification further requires two features:

- Public verification: which permits anyone, not just the clients originally stored the file, to perform verification.
- Stateless verification, which can eliminate the need for state information maintenance at the verifier side between the actions of auditing and data storage.

2) Secure Deduplication: The second outline objective of this work is secure deduplication. In other words, it requires that the cloud server is able to reduce the storage space by keeping only one copy of the same file. Observe that, consider the secure deduplication, our objective is distinguished from previous work in that we propose a method for allowing both deduplication over files and tags.

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

III. SECCLOUD+

We identify that our proposed SecCloud system has accomplished both integrity auditing and file deduplication. Be that as it may, it cannot avoid the cloud servers from knowing the content of files having been stored. In other words, the functionalities of integrity auditing and secure deduplication are only imposed on plain files. In this section, in order to provide confidentiality we propose SecCloud+, which allows for integrity auditing and deduplication on encrypted files.

A. System Model

Contrasted with SecCloud, our improved proposed SecCloud+ involves an additional trusted third party entity, called as key server, which is responsible for providing clients with secret key (according to the file content) for encrypting files. In our proposed work we distinguish with the previous work by allowing for integrity auditing on encrypted data instead of doing on plain files. SecCloud+ follows the same three protocols (i.e., file uploading protocol, the integrity auditing protocol and the proof of ownership protocol) as with SecCloud. The one only one difference is the file uploading protocol in SecCloud+ involves an additional extra phase for communication between cloud client and key server. That is, the client must communicate with the key server to get the convergent key for encrypting the uploading file before the phase 2 in SecCloud. Unlike SecCloud, another design goal of file confidentiality is desired in SecCloud+ as follows.

1) File Confidentiality: The design goal of file confidentiality requires completely avoiding the cloud servers from accessing the content of files. Especially, we require that the goal of file confidentiality needs to be control as well as avoid the "dictionary attack". That is, even the adversaries have the pre-knowledge of the "dictionary" which includes all the possible files; they still cannot recover the target file.

IV. CONCLUSION

Aiming at getting both data integrity and deduplication in cloud, we introduce two systems called 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 presenting a Proof of Ownership protocol and avoiding the leakage of side channel information in data deduplication. Compared with past work, 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.

ACKNOWLEDGMENT

We acknowledge all our gratitude to **Prof. Dr. Nagesh H.R** (Dean and Head of Computer Science and Engineering Department) who has provided facilities to explore the subject with more enthusiasm. This experience will always steer us to do our work perfectly and professionally.

REFERENCES

[1]. 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.
[2]. 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.

[3]. 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.

[4]. 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.

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

[6]. 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.

[7]. 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.

[8]. 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.

[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]. 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.

[11]. 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.

[12]. 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.

[13]. 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.

BIOGRAPHIES

Mr. Lohith Kumar K.K completed his B.E. in Information Science and Engineering in 2014 from MCE an autonomous institution Under VTU Belagavi located at Hassan, Karnataka India. Now he is a M.Tech candidate in computer Science and Engineering in MITE Under VTU Belagavi Karnataka Moodbidri. His current interests include Cloud Computing, Image Processing, Network Security.

Mr. Ashwin Kumar completed his B.E. In Computer Science and Engineering in NMAIT Nitte, and also he pursued his M.tech in NMAIT Nitte, He has a 9 year experience in teaching presently he works as Senior Assistant Professor in MITE Moodbidri which comes under VTU Belagavi Karnataka Moodbidri. His current interests include Cloud Computing, Artificial Intelligence, Cryptography and Network Security and Image Processing.