INTERNATIONAL RESEARCH JOURNAL IN ADVANCED ENGINEERING AND TECHNOLOGY (IRJAET) E - ISSN: 2454-4752 P - ISSN : 2454-4744
VOL 2 ISSUE 2 (2016) PAGES 832-836
RECEIVED : 26/03/2016. PUBLISHED : 04/04/2016
April 4, 2016

SECURE AUDITING AND DEDUPLICATING DATA IN CLOUD

[1]Pavithra.S, [2]Sumithra.R, [3]Dharini.R.K, [4]Swetha.R, [5]Saravanan.K,
[1,2,3,4]Dept of Computer Science and Engineering, Pavai College Of Technology, Namakkal,
[5]HOD, Dept of Computer Science and Engineering, Pavai College Of Technology, Namakkal.

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.

1. 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 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 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.

2. RELATED WORK

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 [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.

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.

INTERNATIONAL RESEARCH JOURNAL IN ADVANCED ENGINEERING AND TECHNOLOGY (IRJAET) E - ISSN: 2454-4752 P - ISSN : 2454-4744
VOL 2 ISSUE 2 (2016) PAGES 832-836
RECEIVED : 26/03/2016. PUBLISHED : 04/04/2016
April 4, 2016

3. SYSTEM MODEL

Aiming at allowing for auditable and deduplicated storage, we propose the SecCloud system. Cloud Clients have large data files to be stored and rely on the cloud for data maintenance and computation. They can be either individual consumers or commercial organizations. Auditor which helps clients 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 made available to the other entities in the system. 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.

Fig.1.Proposed Model

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.

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.

i**)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.

ii)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.

We determine that our proposed SecCloud system has achieved both integrity auditing and file deduplication. However, 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, we propose SecCloud+, which grant for integrity auditing and deduplication on encrypted files. System Model Compared with SecCloud, our recommended SecCloud+ involves further trusted entity, namely key server, which is responsible for assigning clients with secret key (according to the file content) for encrypting files. This architecture is in line with the recent work. But our work is distinguished with the past work by allowing for integrity auditing on encrypted data. SecCloud+ follows the same three protocols (i.e., the file uploading protocol, the integrity auditing protocol and the proof of ownership protocol) as with SecCloud. The only anomaly is the file uploading protocol in SecCloud+ involves an additional stages for communication among cloud client and key server. That is, the client needs to communicate with the key server to get the convergent key for encrypting the uploading file before the phase in SecCloud.

CONCLUSION

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] 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.179194.[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. 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.