RESEARCH ARTICLE OPEN ACCESS

Avoid Data Duplication in Cloud

S.Ramya[1], A.Petrisia[2]
1PG Student, 2Associate Professor
[1, 2]Department of MCA, Dhanalakshmi Srinivasan College of Engineering and Technology

Abstract:

As the cloud computing technology develops during the last decade; outsourcing data to cloud service for storage becomes an attractive trend, which benefits in sparing efforts on heavy data maintenance and management. Nevertheless, since the outsourced cloud storage is not fully trustworthy, it raises security concerns on how to realize data de-duplication in cloud while achieving integrity auditing. In this work, we study the problem of integrity auditing and secure de-duplication on cloud data. Specifically, aiming at achieving both data integrity and de-duplication in cloud, we propose 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. 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 de-duplication on encrypted data.

Keywords — **Cloud Computing, Encryption, Servers.**

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 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. 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. The cloud server is able to relieve clients from the heavy burden of storage management and maintenance. The most 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, 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 de-duplication. 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 recent survey by EMC, 75% of recent digital data is duplicated copies. This fact raises a technology namely de-duplication, in which the cloud servers would like to de-duplicate by keeping only a single copy for each file (or block) and make a link to the file (or block) for every client who owns or asks to store the same file (or block). Unfortunately, this action

of de-duplication would lead to a number of threats potentially affecting the storage system, 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 exact 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). Thus, the second problem is generalized as 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.

2. LITERATURE SURVEY

In [1], J. Yuan and S. Yu discussed about Secure and constant cost public cloud storage auditing with deduplication. Data integrity and storage efficiency are two important requirements for cloud storage. Proof of Retrievability (POR) and Proof of Data Possession (PDP) techniques assure data integrity for cloud storage. Proof of Ownership (POW) improves storage efficiency by securely removing unnecessarily duplicated data on the storage server. However, trivial combination of the two techniques, in order to achieve both data integrity and storage efficiency, results in non-trivial duplication of metadata (i.e., authentication tags), which contradicts the objectives of POW. Recent attempts to this problem introduce tremendous computational and communication costs and have also been proven not secure. It calls for a new solution to support efficient and secure data integrity auditing with storage deduplication for cloud storage. Here they solve this open problem with a novel scheme based on techniques including polynomial-based authentication tags and homomorphic linear authenticators. Their design allows deduplication of both files and their corresponding authentication tags. Data integrity auditing and storage deduplication are achieved simultaneously. Their scheme is also characterized by constant realtime communication and computational cost on the user side. Public auditing and batch auditing are both supported. Hence, their proposed scheme outperforms existing POR and PDP schemes while providing the additional functionality of deduplication.

In [2] S. Keelveedhi discussed about Dupless: Serveraided encryption for deduplicated storage. Cloud storage service providers such as Dropbox, Mozy, and others perform deduplication to save space by only storing one copy of each file uploaded. Should clients conventionally encrypt their files, however, savings are lost. Message-locked encryption (the most prominent manifestation of which is convergent encryption) resolves this tension.

However it is inherently subject to brute-force attacks that can recover files falling into a known set. They propose an architecture that provides secure deduplicated storage resisting brute-force attacks, and realize it in a system called DupLESS. In DupLESS, clients encrypt under message-based keys obtained from a key-server via an oblivious PRF protocol. It enables clients to store encrypted data with an existing service, have the service perform deduplication on their behalf, and yet achieves strong confidentiality guarantees. They show that encryption for deduplicated storage can achieve performance and space savings close to that of using the storage service with plaintext data.

In Proxy provable data possession in public clouds the author discussed about cloud computing rapidly expands as an alternative to conventional computing due to it can provide a flexible, dynamic and resilient infrastructure for both academic and business environments. In public cloud environment, the client moves its data to public cloud server (PCS) and cannot control its remote data. Thus, information security is an important problem in public cloud storage, such as data confidentiality, integrity, and availability. In some cases, the client has no ability to check its remote data possession, such as the client is in prison because of committing crime, on the ocean-going vessel, in the battlefield because of the war, and so on. It has to delegate the remote data possession checking task to some proxy. They study proxy provable data possession (PPDP). In public clouds, PPDP is a matter of crucial importance when the client cannot perform the remote data possession checking. They study the PPDP system model, the security model, and the design method. Based on the bilinear pairing technique, we design an

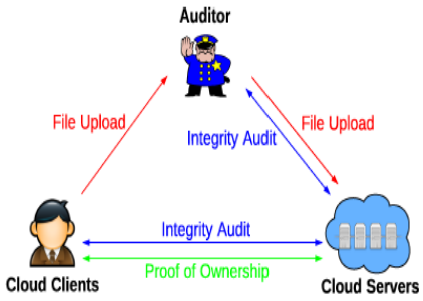
efficient PPDP protocol. Through security analysis and performance analysis, their protocol is provable secure and efficient [3].

3. PROPOSED SYSTEM

The system architecture is shown below in Fig 1.

Fig.1 Overall System Architecture

The



Proposed System consists of five system modules. They are:

- File Uploading Protocol
- Integrity Auditing Protocol
- Proof of Ownership Protocol
- Secure De-duplication
- File Confidentiality

A. 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): 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. 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.

B. 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 proffer, 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 proved (i.e., cloud server).

Phase 2 (cloud server → cloud client/auditor): based on the stored files and file tags, proved (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.

C. 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

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.

D. Secure De-duplication

The main design goal of this work is secure de-duplication. 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. Notice that, regarding to secure de-duplication, our objective is distinguished from previous work in that we propose a method for allowing both de-duplications over files and tags.

E. File Confidentiality

The design goal of file confidentiality requires preventing the cloud servers from accessing the content of files. Specially, we require that the goal of file

confidentiality needs to be resistant to "dictionary attack". That is, even the adversaries have pre-knowledge of the "dictionary" which includes all the possible files, they still cannot recover the target file.

Advantages of the Proposed System

- 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.

- Secure Deduplication

The second design goal of thiswork 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.

- 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.

4. RESULTS

The Home page is shown in Fig.2. First register the details in the Registration form as shown in the figure Fig 3.

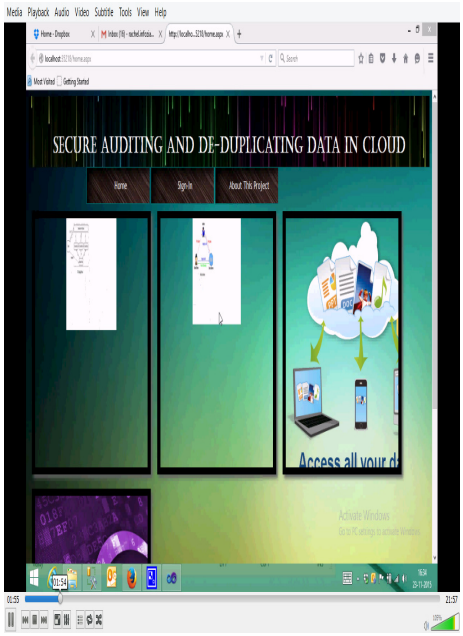


Fig.2 Home Page

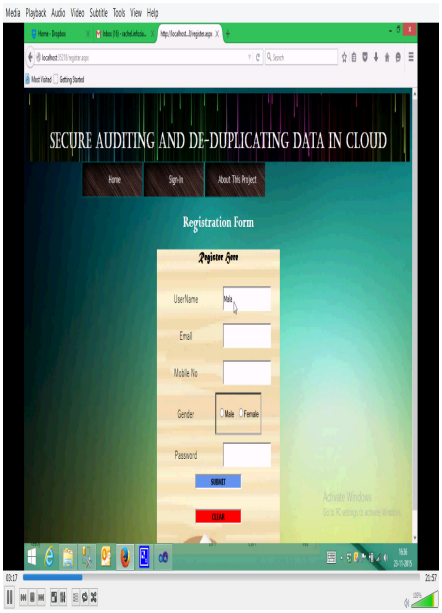


Fig.3 Registration Form Page

The files can be uploaded and then audited as shown in the figure Fig 4. Finally the audited files are stored in the cloud as shown in the figure Fig 5.

Fig.4 Auditing File

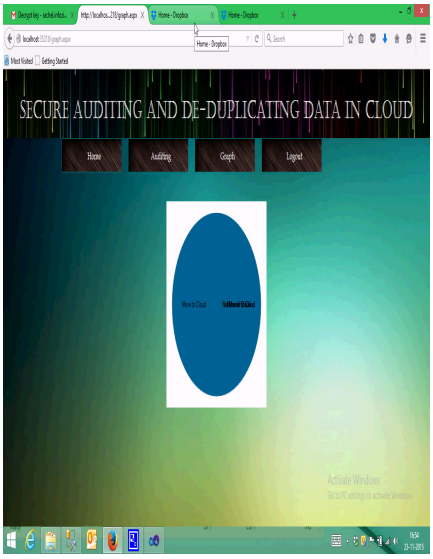


Fig.5 Audited Files Stored in Cloud

5. CONCLUSION AND FUTURE ENHANCEMENT

Aiming at achieving both data integrity and de-duplication in cloud, we propose 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. In addition, SecCoud enables secure de-duplication through introducing a Proof of Ownership protocol and preventing the leakage of side channel information in data de-duplication. Compared with previous 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 de-duplication directly on encrypted data.

REFERENCES

[1] J. Yuan and S. Yu, "Secure and constant cost public cloud storageauditing with deduplication," in IEEE Conference on Communications and Network Security (CNS), 2013, pp. 145–153.

[2] 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.

[3] H. Wang "Proxy Provable Data Possession in Public Clouds" in IEEE Transactions on services computingIssue No.04 - Oct.-Dec. (2013 vol.6)pp: 551-559

[4] 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.

[5] 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.

[6] 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.

[7] 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. 12:1–12:34,20