SECURE CLOUD DATA DEDUPLICATION MECHANISM FOR REMOTE STORAGE

Hema Sable[1], Savita R.Bhosale[2]

[1] *PG Scholar, Computer Engineering, MGMCET Kamothe, Navi Mumbai (India)*

[2]*Professor, Electronics and Telecommunication, MGMCET Kamothe, Navi Mumbai (India)*

ABSTRACT

Data de-duplication techniques is imperative data compression techniques which avoid duplicate copies of redundant data, and it helps to save the bandwidth by diminish the storage space in cloud storage. De-duplication provides the higher security and protects the perceptive data by using the convergent encryption technique which encrypts the data before outsourcing. Data de-duplication extensively used in cloud computing. There is limited source of storage and networking in cloud system. So data de-duplication takes an essential role in the cloud structure. Security scrutiny demonstrates that de-duplication is secure in terms of the definitions specified in the proposed security model. We execute an example of our projected authorized duplicate check method and perform test experiments using our prototype. Which provide minimum overhead compared to normal operations.

Keywords: Deduplication, Authorized Duplicate Check, Confidentiality, Cloud Computing, Convergent Encryption

I. INTRODUCTION

Best use of cloud computing is utilization of enormous "virtualized" resources across the whole internet along with all security measures like platform and implementation details hiding. Cloud service providers propose potentially infinite storage space at reasonable costs [1]. One critical challenge of cloud storage services is the managing of the mounting volume of data. Data de-duplication is the removal of replicate data. In the de-duplication process, duplicate data is deleted, only one copy or single occurrence of the data to be stored in the database [2]. Important issues in data deduplication that security and privacy to protect the data from insider or outsider attack. For data confidentiality, encryption is used by different user for encrypt their files or data, using secrete key user perform encryption and decryption operation. Traditional encryption requires different users to encrypt their data with their own keys. Convergent encryption has been proposed to enforce data confidentiality while making deduplication feasible. To prevent unauthorized access proof of ownership protocol is used to present proof that the user really owns the same file when deduplication found. After the proof, server provides a pointer to consequent user for accessing same file without needing to upload same file. When user want to download file he simply download encrypted file from cloud and decrypt this file using convergent key[3,4,5,6,7,8,9,10].

II. PRELIMINARIES

In this paper, we use some secure primitives which are used in our secure deduplication.

2.1 Symmetric Encryption

Symmetric encryption uses a common secret key κ to encrypt and decrypt information.

A symmetric encryption scheme consists of three primitive functions:

- KeyGenSE (1)! K is the key generation algorithm that generates κ using security parameter 1_;
- EncSE (κ, M)! C is the symmetric encryption algorithm that takes the secret κ and message M and then Outputs the ciphertext C; and
- DecSE (κ, C)! M is the symmetric decryption algorithm that takes the secret κ and ciphertext C and then outputs the original message M.

2.2 Convergent Encryption

Convergent encryption provides data confidentiality in deduplication. A convergent encryption scheme can be defined with four primitive functions:

- KeyGenCE (M)! K is the key generation algorithm that maps a data copy M to a convergent key K;
- EncCE (K, M)! C is the symmetric encryption algorithm that takes both the convergent key K and the data copy M as inputs and then outputs a ciphertext C;
- DecCE (K, C)! M is the decryption n algorithm that both the ciphertext C and the convergent t key K as inputs and then outputs the original data copy M; and
- TagGen (M)! T (M) is the tag generation algorithm that maps the original data copy M and outputs a tag T (M).

2.3 Proof of Ownership

The Proof of Ownership (PoW) is challenge-response protocol enabling a storage server to check whether a requesting entity is the data owner, based on a short value. That is, when a user wants to upload a data file (D) to the cloud, he first computes and sends a hash value hash = H (D) to the storage server. This latter maintains a database of hash values of all received files, and looks up hash. If there is a match found, then D is already outsourced to cloud servers. As such, the cloud tags the cloud user as an owner of data with no need to upload the file to remote storage servers. If there is no match, then the user has to send the file data (D) to the cloud. This client side deduplication, referred to as hash-as-a proof presents several security challenges, mainly due to the trust of cloud users assumption. This Section presents a security analysis of existing PoW Schemes Despite the significant resource saving advantages, PoW schemes bring several security challenges that may lead to sensitive data.

III. PROPOSED SYSTEM DESCRIPTION

Our secure client-side data deduplication mechanism is based on an creative use of the convergent encryption. That is, on one hand, when a data owner wants to store a new enciphered data file in remote storage servers, he/she has first to generate the enciphering key. This data encrypting key is derived by applying a one way hash function on data content. After successfully encrypting the file data, the client has to generate the data identifier of enciphered data, in order to check its uniqueness in cloud database, before uploading the claimed file. This data identifier is computed by using a Merkle hash tree, over encrypted contents. Then, for subsequent data outsourcing, the client is not required to send the same encrypted data. However, he has to substitute a client-server interactive proof scheme (PoW), in order to prove his ownership

On the other hand, to protect data in public cloud servers from unauthorized entities, the client has to ensure that only authorized users are able to obtain the decrypting keys. As such, the data owner has to encrypt the data deciphering key, using the public key of the recipient user. This key is, then, integrated by the data owner in user metadata, ensuring data confidentiality against malicious users, as well as flexible access control policies. To illustrate our solution for improving data security and efficiency, we first present the different prerequisites and assumptions. Then, we introduce three use cases for storing, retrieving and sharing data among a group of users.

3.1 Assumptions

Our solution considers the following assumptions. First, we assume that there is an established secure channel between the client and the CSP. This secure channel supports mutual authentication and data confidentiality and integrity. Hence, after successfully authenticating with the CSP, these cloud users share the same resources in a multi-tenant environment. Second, our solution uses the hash functions in the generation of the enciphering data keys. Hence, we assume that these cryptographic functions are strongly collision resistant, as it is an intractable problem to find the same output for different data files.

3.2 Prerequisites

• Merkle Hash Tree – a Merkle tree MT provides a succinct commitment, called the root value of the Merkle tree, to a data file. That is, the file is divided into blocks; called tree leaves, grouped in pairs and hashed using a collision resistant hash function. The hash values are then grouped in pairs and the process is repeated until the construction of the root value. The Merkle tree proof protocol requires that the prover has the original data file. That is, the verifier chooses a number of leaf indexes and asks the verifier to provide the corresponding leaves. As such, the verifier has to send these leaves with a sibling valid path.

• Interactive Proof System– this proof system is an interactive game between two parties: a challenger and a verifier that interact in a common input, satisfying the correctness properties (i.e. completeness and soundness).

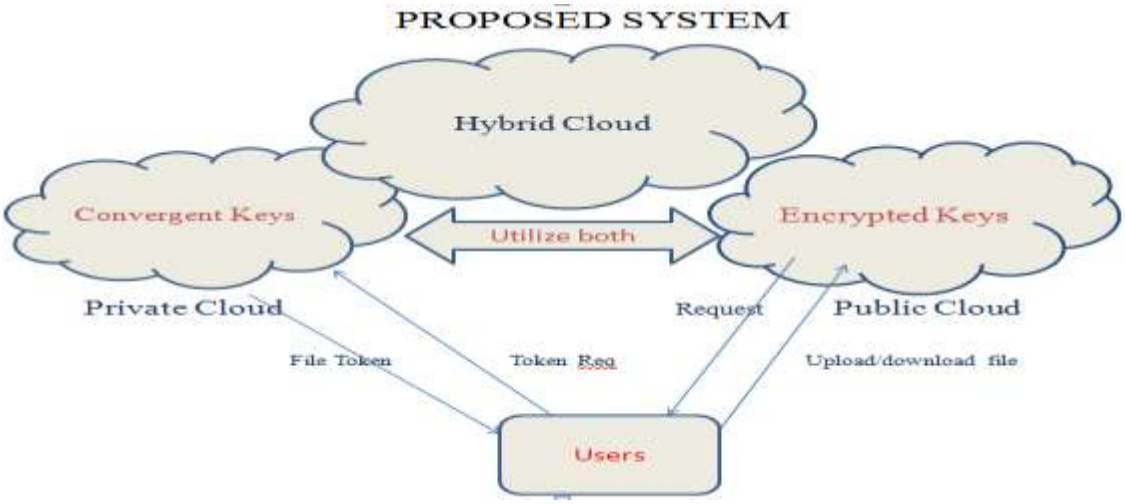

Figure 1: Architecture of the Authorised Deduplication

It is not possible to provide security to the private data by using only the public cloud so the data will loss. Figure 1 shows the authorised deduplication which provides data security by using private cloud. This system provides data deduplication, which saves the bandwidth and eliminate the duplicate copies of the data. User can upload and download the files from public cloud but private cloud provides the security for that data. Only

authorized person can upload and download the files from the public cloud by generating the convergent key. Keep that key into the private cloud. When user request to the private cloud for downloading key and then accesses that Particular file.

IV. JUDGEMENT

We conduct assessment on our model. Our project aim is to diminish the overhead generated by authorization steps, which include file token generation and share token generation, against the convergent encryption and file upload steps. We assess the overhead by changing different factors,

1) File Size
2) Number of Stored Files
3) Deduplication Ratio
4) Privilege Set Size

We perform the experiments with three machines equipped with an Intel Core-2-Quad 2.66Ghz Quad Core CPU, 4GB RAM and installed with Ubuntu 12.04 32- Bit Operation System. The machines are connected with 1Gbps Ethernet network. We break down the upload process into 6 steps, 1) Tagging 2) Token Generation 3) Duplicate Check 4) Share Token Generation 5) Encryption 6) Transfer. For each step, we trace the start and end time of it and therefore obtain the breakdown of the total time spent. We present the average time taken in each data set in the figures.

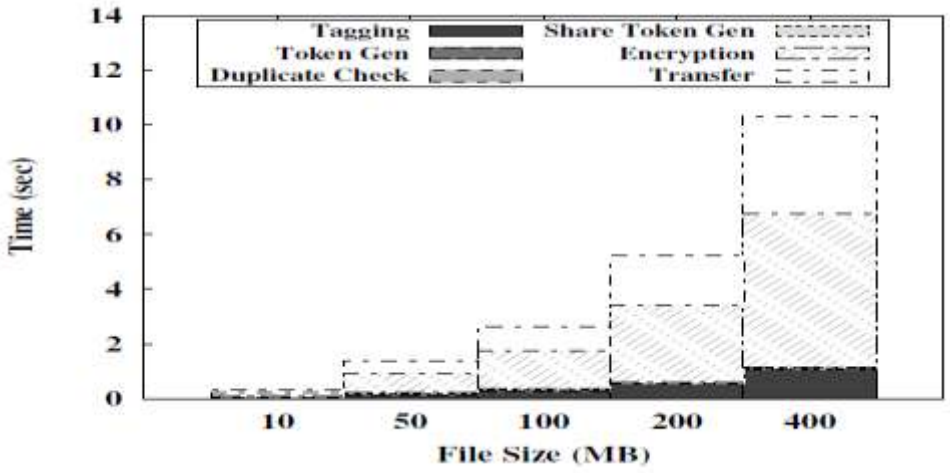

Figure 2: Time Breakdown for Different File Size

4.1 File Size

To assess the outcome of file size to the time spent on different steps, we upload 100 unique files (i.e., without any de-duplication opportunity) of particular file size and record the time break down. Using the unique files enables us to assess the worst-case scenario where we have to upload all file data. The time spent on tagging, encryption, upload increases linearly with the file size, since these operations involve the actual file data and incur file I/O with the whole file. In contrast, other steps such as token generation and duplicate check only use the file metadata for computation and therefore the time spent remains constant. With the file size increasing from 10MB to 400MB, the overhead of the proposed authorization steps decreases from 15% to 0.5%.

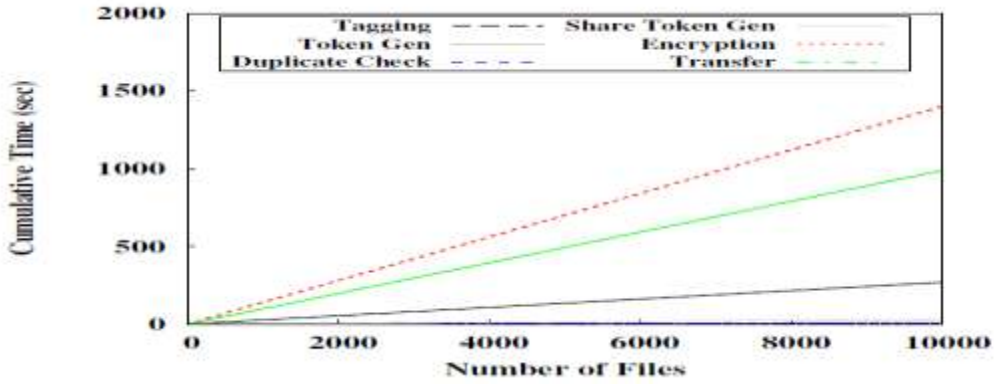

Figure 3:Time Breakdown for Different Number of Stored Files

4.2 Number of Stored Files

To assess the effect of number of stored files in the model, we upload 10000 10MB unique files to the system and record the breakdown for every file upload. From Figure 3, every step remains constant along the time. Token checking is done with a hash table and a linear search would be carried out in case of collision. Despite of the possibility of a linear search, the time taken in duplicate check remains stable due to the low collision probability.

4.3 Deduplication Ratio

To assess the effect of the deduplication ratio, we arrange two unique data sets, each of which consists of 50 100MB files. We first upload the first set as an initial upload. For the second upload, we pick a portion of 50 files, according to the given deduplication ratio, from the initial set as duplicate files and remaining files from the second set as unique files. The average time of uploading the second set is presented in Figure 4. As uploading and encryption would be skipped in case of duplicate files, the time spent on both of them decreases with increasing deduplication ratio. The time spent on duplicate check also decreases as the searching would be ended when duplicate is found. Total time spent on uploading the file with deduplication ratio at 100% is only 33.5% with unique files.

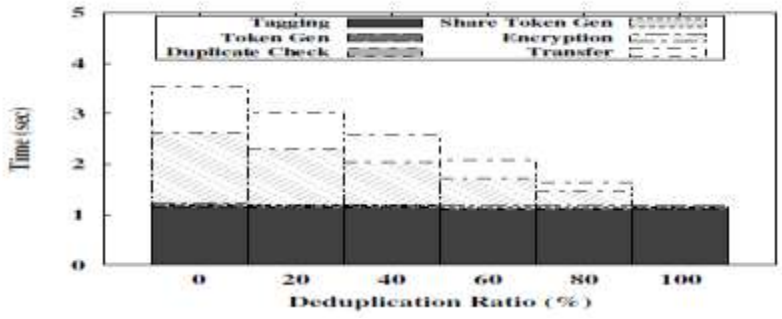

Figure 4: Deduplication Ratio

4.4 Privilege Set Size

To assess the effect of privilege set size, we upload 100 10MB unique files with different size of the data owner and target share privilege set size. In Figure 5, it shows the time taken in token generation increases linearly as more keys are associated with the file and also the duplicate check time. While the number of keys increases 100 times from 1000 to 100000, the total time spent only increases to 3.81 times and it is noted that the file size of the experiment is set at a small level (10MB), the effect would become less significant in case of larger files.

Time taken for the first week is the highest as the initial upload contains more unique data. Overall, the results are consistent with the prior experiments that use synthetic workloads.

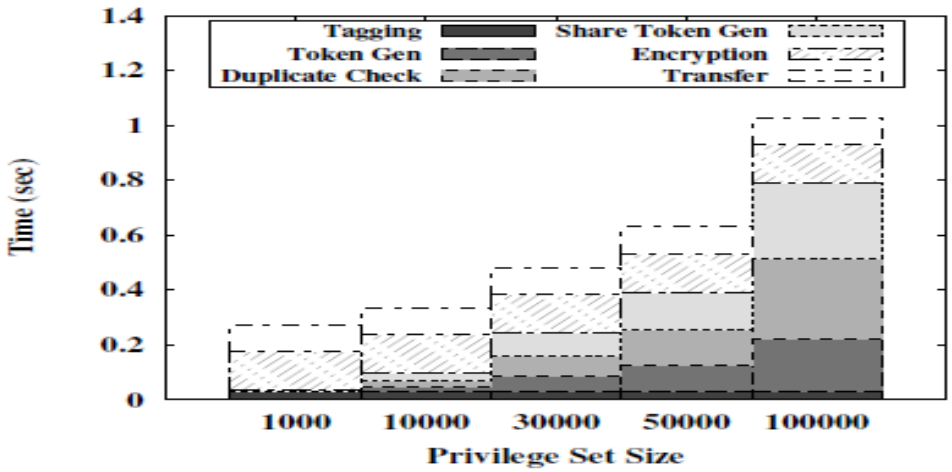

Figure 5: Time Breakdown for Different Privilege Set

V. CONCLUSION

In this paper, the aim of authorised data deduplication is use to defend the data security by including differential privileges of users in the duplicate check. The hybrid cloud architecture is proposed to support the authorised duplicate-check. In which private cloud generate the duplicate check tokens of files, so that our system has been protected from the insider and outsider attacks. And this authorised duplicate check model eliminate overhead than the convergent and network transfer.

VI. ACKNOWLEDGEMENT

I am Hema Sable and I greatly thankful to Dr. Savita R.Bhosale for her guidance. We also thank the college authorities, PG coordinator and Principal Sir for providing the required infrastructure and support. Finally, we would like to extend a heartfelt gratitude to M.E. Staff and friends.

REFERENCES

[1]. Masayuki Okuhara, Tetsuo Shiozaki, Takuya Suzuki " Security Architectures for Cloud Computing " Fujitsu sd. Tech J vol 46, no 4 (Octomber 2010).

[2]. Wee Keong Ng, Yonggang Wen, Huafei Zhu " Private Data Deduplication Protocols in Cloud Storage " march 2012.

[3]. Chao Yang, Jian Ren and Jianfeng Ma " Provable Ownership of File in De-duplication Cloud Storage " IEEE 2013.

[4]. Deepak Mishra, Dr. Sanjeev Sharma " Comprehensive study of data de-duplication" International Conference on Cloud, International Conference on Cloud, Big Data and Trust 2013, Nov 13-15, RGPV.

[5]. Shai Halevi, Danny Harnik, Benny Pinkas, and Alexandra Shulman-Peleg " Proofs Of Ownership in Remote Storage Systems" 2013.

[6]. Mihir Bellare, Sriram Keelveedhi, Thomas Ristenpart " DupLESS: Server - Aided Encryption for Deduplicated Storag " 2013.

[7]. Mihir Bellare, Sriram Keelveedhi , Thomas Ristenpar t " Message - Locked Encryption and Secure Deduplication " March 2013.

[8]. Pasquale Puzio, Molva, Melek O ¨ nen, Sergio Loureiro " Block – level De-Duplication with Encrypted Data " Open Journal of Cloud Computing (OJCC) Volume 1.

[9]. Rajashree Shivshankar Walunj, Deepali Anil Lande, Nilam Shrikrushna Pansare " Secured Authorized Deduplication Based Hybrid Cloud " The International Journal Of Engineering and Science (IJES) Volume 3, Issue 11, Pages 34 - 39 , 2014.

[10]. Pasquale Puzio, Refik Molva, Melek O ¨ nen, Sergio Loureiro " ClouDedup : Secure Deduplication with Encrypted Data for Cloud Storage " Year 2014.