Optimal Data Deduplication In Cloud With Authorization

Divyesh Minjrola
PG Student, Dept of computer engineering
Alard college of engineering and management
Pune, India
Divyesh.minjrola@gmail.com

Sonali Patil
Professor, Dept of computer engineering
Alard college of engineering and managemenet
Pune, India

***Abstract*:-**Cloud technology is widely used technology as it allows sharing and centralized storage of data, sharing of data processing and online access of computer services and resources on various types of devices. One of the critical challenges of cloud storage services is the management of the ever-increasing volume of data .To address these data deduplication is one of the novel technique. Deduplication helps to remove and prevent from having duplicate copies of same data. Though deduplication has several benefits it adds concerns related to privacy and security of users as it can lead to insider or outsider attacks. Achieving deduplication along with data security in cloud environment makes it more critical problem to solve.

Objective of this paper on Optima Authorized Data Deduplication in Cloud is to mention the proposed system and analysis of deduplication techniques and optimal authorization measures for security along with deduplication technique in cloud environment

Keywords- Cloud computing, Deduplication, Convergent Encryption, Proof of Ownership Protocol, Differential Authorization

*****

I. Introduction

Cloud technology is internet based technology, which provides services to user by providing access to shared and centralized storage of data and also processing capabilities in platform independent manner at low cost. As cloud technology becomes prevalent along with it the data storage and sharing also became prevalent. One of the critical challenge of cloud storage services is the management of the increasing volume of data. To address this deduplication is novel technique but it also adds privacy concerns.To make data management scalable in cloud computing, deduplication has been well known technique and this has attracted more and more attention recently. Data deduplication is a special data compression technique, it helps eliminating duplicate copies of repeating data in storage. Deduplication technique is used to improve storage utilization and it is also applied to network data transfers to reduce the number of bytes that must be sent.

In deduplication instead of keeping multiple data copies with the same content, deduplication eliminates redundant data by keeping only single physical copy and referring other redundant data to that single copy. Deduplication can take place at file level or deduplication can be at block level. In file level deduplication, it eliminates duplicate copies of the same file. Deduplication can also be applied at the block level, which eliminates duplicate blocks of data that occur in non-identical files.

Cloud computing service model is highly emerging and it provides computation and storage resources on the Internet. One of the attractive functionality that cloud computing offers is cloud storage. Individuals and organizations often required to remotely archive their data to avoid any information loss in case there are any hardware/software failures or unforeseen disasters. Instead of purchasing the needed storage media to keep data backups, individuals and organizations can outsource their data backup services to the cloud service providers, which provide the necessary storage resources to host the data backups. While cloud storage is attractive, but how to provide security guarantees for outsourced data becomes a rising concern .Although data deduplication has a lot of benefits, security and privacy concerns arise as user sensitive data are susceptible to both insider and outsider attacks

II. Literature Survey

Data deduplication [2] technology is method for maximizing the usage of available data storage. Deduplication helps to identify similarities among different files to save disk space.

Data deduplication inspects data down to block-leveland bit-level and, after the initial occurrence, only the changed data they find are saved. The rest are discarded and replaced with a pointer to the previously saved information. Block-level and bit-level deduplication methods are able to achieve compression ratios of 20x to 60x, or sometime even higher, under the right conditions.

There is file-level deduplication, called single instance storage in file-level deduplication, in this if two files are identical, one copy of the file is kept while subsequent iterations are not. File-level deduplication is not as efficient as block-level and bit-level storage because even a single changed bit results in a new copy of the whole file being stored. For the purposes data deduplication is defined as operating at block level and bit level.

Data deduplication reduces the amount of data that is needed to be stored. This means that less media has to be bought and it takes long to fill up disks and tapes. Data can be backed up more quickly, which means shorter backup windows and quicker restores time. A reduction in the amount of space taken up in disk systems and VTLs [13] for example, longer retention periods are possible, bringing quicker restores to users direct from disk and reducing dependence on tape storage and its management. Less data also means less bandwidth taken up, which means data deduplication can speed up remote backup, replication and also disaster recovery processes.

One of the advantage of deduplication is to minimize storage cost by storing more data on disks.

Deduplication helps to reduce input/output ratio and helps to lower the cost of storage and energy.

Deduplication also he lps to recover files or entire system at certain point in time.

In case of deduplication on files, complete file is used for validation if any other file with similar data is present or not. If similar copy of file is found then another copy of same file is not stored.

Advantage of file level deduplication is it needs less metadata information and it is comparatively easy to implement and maintain.

In chunk level deduplication, file is divided into chunk of same sizes or various sizes. During deduplication, each of this chunk is used for duplicate validation.

If similar chunk (of same or other file) is found then deduplication will only stores a reference to this chunk instead of its actual contents [Figure 1].

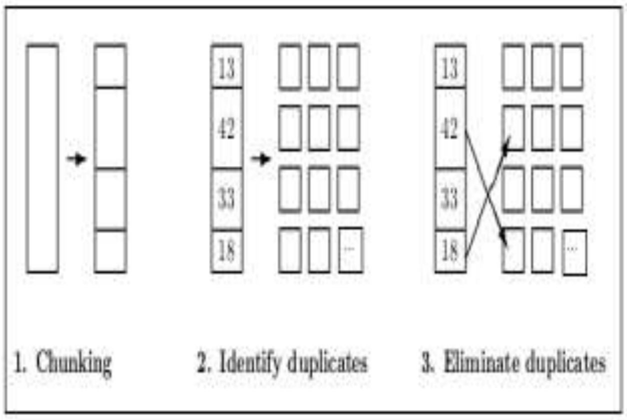


Figure 1: Deduplication for chunks of a file

Conventional or traditional encryption does not work for deduplication as user will encrypt their data with their individual keys and thus the identical of copy of data will have different cipher text and deduplication is impossible.

Convergent encryption [3] is a widely used technique, it enables to combine the storage savings of deduplication while still providing privacy guarantees for the data. In convergent encryption, the data/message is encrypted under a key derived by the hashing the message itself, along with a public parameter.

For example, to encrypt M, we could get K ← H(P, M) and derive C ← Enc(K, M, IV_0). Here, P is a 128-bit string picked at random and fixed throughout the system, IV_0 is the all-0 initialization vector, H could implemented through SHA256 and Enc through AES128 in CTR mode.

Message-locked encryption is a cryptographic primitive that broadens convergent encryption and captures the properties needed for enabling deduplication over ciphertexts

However, convergent encryption (and message-locked encryption in general) is inherently subject to brute-force attacks that can recover files falling into a known set. If a file M is picked from a set S, given a convergent encryption ciphertext C of M, an attacker can trial-decrypt with each candidate file M' in S by deriving K ← H(P, M') and checking if M' = Dec(K, C).

In typical storage system with deduplication, first client will only send the hash value of the file then server to check if that hash value already exists in its database. If file is already present on the server storage it asks client not to send the file and marks client additional owner of the file. Thus client side deduplication leads to security problem and also could reveal other client has same file of sensitive information. Such a problem can be addressed by proof of ownership protocol (PoW)[4]. PoW is in two parts and its between two players on common input file. In step one it verifier summaries to itself and generate sort information v. In second step prover and verifier engage in interactive protocol where verifier has sort information v and prover has file F at the end verifier either accepts or rejects it.

Cloud storage systems are becoming increasingly popular over the time. A promising technology that helps to keep their cost down is deduplication, which stores only a single copy of repeating data. Client-side deduplication[8]technique attempts to identify deduplication opportunities already at the client and save the bandwidth of uploading copies of existing files to the server. More specifically, an attacker who knows the hash signature of a file can convince the storage service that it owns that file, hence the server can let the attacker download the entire file. (In parallel to our work, a subset of these attacks was recently introduced in the wild with respect to the Dropbox file synchronization service.) To overcome such attacks, proofs-ofownership (PoWs) is used, which lets a client efficiently prove to a server that that the client holds a file, rather than just some short information about it.

Limitation with PoW protocol is it cannot support differential authorization duplicate check, which useful in many applications.

The Farsite distributed file system provides the availability by replicating each file onto multiple desktop computers. This replication consumes significant storage space and it is important to reclaim used space where possible. The result of measurement of over 500 desktop file systems shows that nearly half of all consumed space is occupied by duplicate files. Reclaiming Space from Duplicate Files in a Serverless Distributed File System [5] present a mechanism that reclaim space from this incidental duplication to make it available for controlled file replication. This mechanism includes 1) convergent encryption, which enables duplicate files to coalesced into the space of a single file, even if the files are encrypted with different users keys, and 2) SALAD, a Self-Arranging, Lossy, Associative Database for aggregating file content and location information in a decentralized, scalable, fault-tolerant manner.

Extreme Binning [15] is a scalable deduplication technique for non-traditional backup workloads that are made up of individual files with no locality among the consecutive files in a given window of time. Due to lack of locality, the existing techniques perform poorly on these workloads. Extreme Binning technique exploits file similarity instead of locality, and makes only one disk access for chunk lookup per file, which gives reasonable throughput. Multi-node backup systems which are built with Extreme Binning scale gracefully with the amount of input data; more backup nodes can be added to boost throughput. Each file is allocated using a stateless routing algorithm to only one node, allowing for maximum parallelization, each backup node is autonomous with no dependency across nodes and making data management tasks robust with low overhead.

III. System Architecture and Design

A. *Problem Defination*

Data deduplication is one of important data compression techniques for eliminating duplicate copies of repeating data, and deduplication is used in cloud storage to reduce the need for amount of storage space and it's also used to save bandwidth. To protect the confidentiality of sensitive data while supporting deduplication the techniques like convergent encryption technique has been proposed and used to encrypt the data before outsourcing. This paper makes the attempt to formally address the problem of authorized data deduplication.

Here we aim at efficiently solving the problem of handling huge volume of data by chunk level deduplication with differential privileges in cloud computing, we consider a hybrid cloud architecture consisting of a public cloud and a private cloud

B. *System Architecture*

Proposed system also has three entities user, private cloud and public cloud.

User: In this system user is an entity who wants to store data on cloud. Each user in system is assigned set of privileges for example, we may define a role based privilege [9] or according to job positions (example. Manager, technical lead and engineer), or define a time-based privileges that indicate validity time period. A user, say Ramesh, may be assigned two privileges technical lead and access right valid till "2015-11-10, so that Ramesh can access any file whose access role is technical lead and accessible till "2015-11-10". In system with deduplication will not upload any duplicate data to save bandwidth and storage.

Private Cloud: This is a new entity for facilitating users secure use of cloud services. The private keys for user privileges are managed by private cloud, which provides the file token to users. The interfaces offered by the private cloud allows user to submit files and queries to be securely stored and computed respectively.

Public cloud: This is an entity that provides the data storage service. To reduce storage cost public cloud reduces redundant data via deduplication.

The flow of operation is mention in [Figure 2]

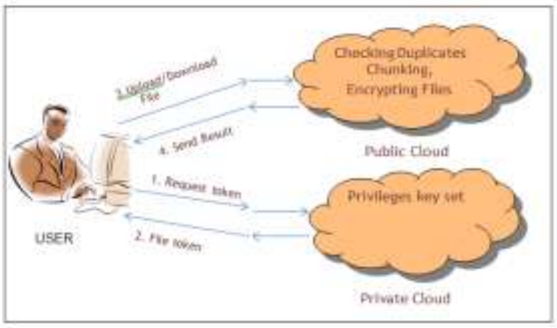


Figure 2: Flow of operation in "Optimal authorized deduplication in Multi cloud system"

Unlike existing system, the proposed system used chunking method of deduplication. To save storage space, deduplication references redundant data instead of copying its contents. This concept relies on that each individual chunk has its own identity, i.e. an identifier by which it can be unambiguously referenced. For each chunk of a file, this identifier is calculated and compared to the global chunk index. If a chunk with a matching identifier is found in the database, the two chunks are assumed to be identical and the existing chunk is referenced.

In case of chunk level deduplication, file can be divided into chunk of same sizes or various sizes. During deduplication, each chunk is used for validation.

If similar chunk (of same or other file) is found then deduplication only stores a reference to this chunk instead of its actual contents

C. *Algorithm and Mathematical model of system*

Algorithm

Input: Data file to upload

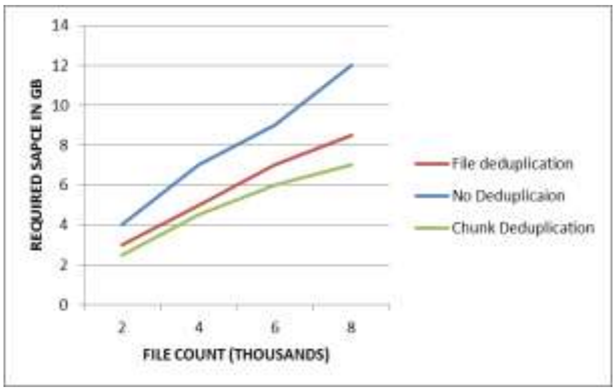
Steps:

1. Take d = data file as a input file.
2. Check if entire file is duplicate
3. Else apply Chunking to the data file
4. Divided chunks c1,c2,c3…cN. Provide Signature to each chunk.
5. Check for duplicate chunks using bloom filter
6. Upload unique chunks

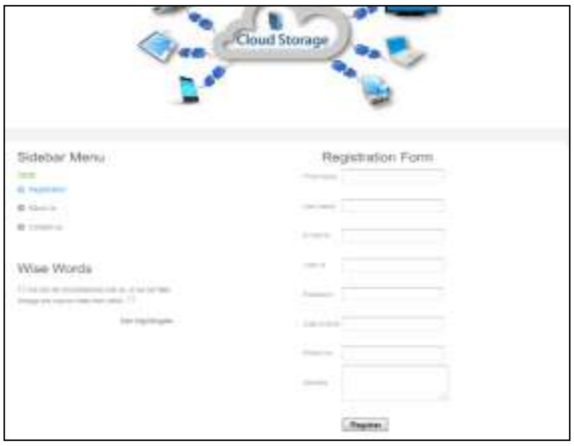
D. *Results and implementation*

We did POC (proof of concept) study by taking data stored in personal computer from 7 employees working in group in an organization.
In POC we considered few thousand files and analyzed space requirement for these data, space saving with file level deduplication and space saving with chunk level deduplication.
Space saving from file level deduplication is around 25% whereas chunk deduplication has spacing of around 37%

Below mentioned graph [Figure 3] demonstrates POC results.

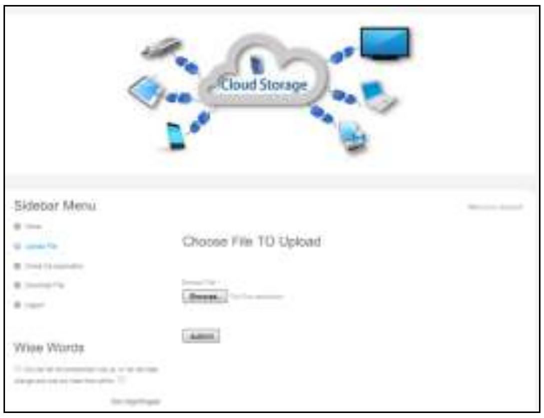


User Registration: This is user registration window. User need to register once in order to use cloud.

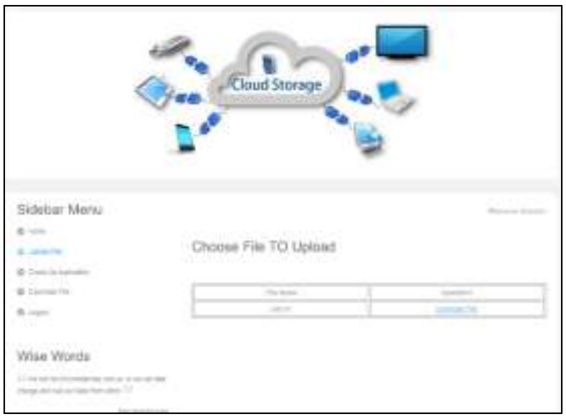


***File Upload:* This window allows user to browse file from computer and upload to cloud.**

When user upload files the system check for duplicate file in database. If file is present then system will maintain reference for existing file and will avoid duplicate. This operation is abstract from user.



***File Download:* This window allows user to download previously uploaded files.**



IV. ACKNOWLEDGMENT

I express my sincere and profound thanks to my project Guide for his consistence support and valuable suggestions, my Head Of Department who always stood as the helping and guiding support for me throughout the delivery of this Project

V. REFERENCES

[1] Jin Li, Yan Kit Li, Xiaofeng Chen, Patrick P. C. Lee, Wenjing Lou, A Hybrid Cloud Approach for Secure Authorized Deduplication, IEEE Transactions on Parallel and Distributed Systems, 2014
[2] J. R. Douceur, A. Adya, W. J. Bolosky, D. Simon, and M. Theimer. Reclaiming space from duplicate files in a serverless distributed file system. In ICDCS, pages 617–624, 2002.
[3] S. Halevi, D. Harnik, B. Pinkas, and A. Shulman-Peleg. Proofs of ownership in remote storage systems. In Y. Chen, G. Danezis, and V. Shmatikov, editors, ACM Conference on Computer and Communications Security, pages 491–500. ACM, 2011
[4] S. Quinlan and S. Dorward. Venti: a new approach to archivalstorage. In Proc. USENIX FAST, Jan 2002
[5] P. Anderson and L. Zhang. Fast and secure laptop backups withencrypted de-duplication. In Proc. of USENIX LISA, 2010
[6] M. Bellare, C. Namprempre, and G. Neven. Security proofs for identity-based identification and signature schemes. J. Cryptology, 22(1):1–61, 2009.
[7] M. Bellare and A. Palacio. Gq and schnorr identification schemes: Proofs of security against impersonation under active and concurrent attacks. In CRYPTO, pages 162–177, 2002
[8] R. S. Sandhu, E. J. Coyne, H. L. Feinstein, and C. E. Youman.Role-based access control models. IEEE Computer, 29:38–47, Feb 1996.
[9] D. Ferraiolo and R. Kuhn. Role-based access controls. In 15th NIST-NCSC National Computer Security Conf., 1992.
[10] J. Li, X. Chen, M. Li, J. Li, P. Lee, andW. Lou. Secure deduplication with efficient and reliable convergent key management. In IEEE Transactions on Parallel and Distributed Systems, 2013.
[11] J. Stanek, A. Sorniotti, E. Androulaki, and L. Kencl. A secure data deduplication scheme for cloud storage. In Technical Report, 2013.
[12] M. W. Storer, K. Greenan, D. D. E. Long, and E. L. Miller. Secure data deduplication. In Proc. of StorageSS, 2008
[13] J. R. Douceur, A. Adya, W. J. Bolosky, D. Simon, and M. Theimer, Reclaiming space from duplicate files in a serverless distributed file system, in Proc. ICDCS'02, 2002, pp. 617-624.
[14] D. Harnik, B. Pinkas, and A. Shulman-Peleg, Side channels in cloud services: deduplication in cloud storage, IEEE Security & Privacy, vol. 8, no. 6, Nov.-Dec. 2010, pp.40-47
[15] U Karthik A Dynamic Load Balancing Algorithm in Computational Grid Using Fair Scheduling, IJCSI International Journal of Computer Science Issues, Vol. 8, Issue 5, No 1, September 2011
[16] Kave Eshghi, Darrell D. E. Long Extreme Binning: Scalable, Parallel Deduplication for Chunk-based File Backup, . IEEE Computer, 29:38–47, Feb 2004