Block Level Data Deduplication for Faster Cloud Backup

Vikram V. Badge, Mrs. Rushali A. Deshmukh

Abstract— Easy access of information and data on the internet enable the users download and store lots of data in their system. Most of storage space is occupied by the duplicate data that have been downloaded or gathered from other resources and stored in the system thus increasing the requirement of the storage space. While taking the backup to the cloud same redundant data is uploaded to the cloud storage. With increase in the amount of such redundant data proper utilization of the storage resources and bandwidth is not possible. Once the data is uploaded at the cloud the user's are not sure how secure their data is. In this paper block level deduplication approach is applied to reduce the data redundancy. In order to maintain the secrecy of the data SHA-256 a cryptographic hash algorithm is implemented. Also compression is performed on the data so that proper bandwidth utilization is possible.

Keywords— Data deduplication; Data compression; Indexing; cloud; SHA-256; Collision resistant.

1. Introduction

Major IT resources are utilized by IT organizations in order to provide the customers with different types of services and backup facility. The easy access of information from the Internet and other resources the redundant data is increasing. The user downloads the similar type of data many times, thus the redundant data is occupying more disk space. The backup of such duplicate data is taken on the cloud to get an instant access when ever required. Thus different challenges arise in cloud backup services [2]. One of the challenge is large backup window, due to the low network bandwidth between user and service provider constraining the data transmission. Also increasing number of data chunks increases the number of fingerprint to be stored which results in hash collision.

The redundancy of the data at the cloud storage is increasing. Thus exploiting the duplicate data can help in saving the backup space. It also helps in reducing the time required for backups in most cases [5]. File level deduplication and block level deduplication [14].

To reduce the duplicate data, deduplication is performed on the data of which backup is taken. The data deduplication is categorized in to two types: In file level deduplication duplicate files are eliminated, it is also referred as Single instance storage. This type of deduplication is not very efficient. In Block level deduplication files are separated in to small blocks of data called chunks, these chunks are identified by their fingerprint [1][7]. The fingerprint is the cryptographic hash of the chunk data. The data deduplication replaces the duplicate chunks with their fingerprints after chunk fingerprint index lookup and only transfers or stores the unique chunks for the purpose of communication or storage efficiency. The block level deduplication approach discussed in the paper reduces the duplicate data at the client side and the cloud side. The chunks of data are formed at the client side and fingerprints of this data chunks are generated by using the hashing techniques. These fingerprints of the chunks are compared with the fingerprints of chunks present at the client side. By this the similar data present during uploading process can be excluded. Data deduplication not only reduces the storage space requirement and but also reduces the duplicate data to be transferred over the network [9].

This paper is composed further as: Section II discuss about related work on the data deduplication and the data compression. Section III problem statement is discussed. Section IV describes system architecture, Mathematical model and Algorithm of the data deduplication system. Section V shows the evaluation of the system. Section VI discussion is performed on the system. Section VII discusses the Conclusion and the future scope of the system.

2. Related Work

In [8] Cumulus system the file system backups could be efficiently implemented over the Internet. This system was

- *Vikram V. Badge is currently pursuing masters degree program in Computer Engineering from JSPM's Rajarshi Shahu college of Engineering, Savitribai Phule Pune University, Pune, India, His area of research is Data mining. E-mail: badgevikram@gmail.com*
- *Prof. Rushali A. Deshmukh pursuing PhD , completed M.E. & B.E. in Computer Science & Engineering. Her key research include NLP, Cloud computing, Data mining. She is currently working as Assistant Professor in JSPM's Rajarshi Shahu college of Engineering, Phule Pune University, Pune, India , Department of Post Graduate Computer engineering with the total Experience of about 15 years. E-mail: radesh19@gmail.com*

specifically designed under a thin cloud, assuming that the remote datacenter storing the backups does not provide any special backup services. In [2] a hybrid source deduplication scheme is designed, it combines file level and chunk level deduplication scheme based on file semantics to reduce lookup overhead by reducing metadata. In paper [1] Application aware Local and Global source deduplication system is introduced, this system combines local source deduplication and global source deduplication. The Duplicate chunks are identified by comparing the fingerprint at the local and global source. The data chunks in different size are sent over the network, thus the chunk size occupy more bandwidth. In [4] the existence of duplicate files is determined from the metadata. The files are clustered into bins depending on their size. They are then segmented, deduplicated, compressed and stored. Binning restricts the number of segments and their sizes so that it is optimum for each file size. When the user requests a file, compressed segments of the file are sent over the network along with the file-to-segment mapping. SHA-1 hash algorithm is more collision resistant than MD5 hash algorithm [1]. In [14] SHA-256 algorithm is implemented has the less probability of hash collision as compared to SHA-1.

The traditional backup solutions require a rotational schedule of full and incremental backup, which move a significant amount of redundant data every week [10]. Most organizations also create a second copy of this information to be shipped to a secondary site for disaster recovery purposes. Thus aggregating, the costs of traditional backup in terms of bandwidth, storage infrastructure, and time increases the cost of IT organizations for information management. Backing up of redundant files and data increases the backup window size, this results in over utilization of network resources and require too much additional storage capacity to hold unnecessary backup data.

Internet bandwidth is the main challenge for backup services, as Internet bandwidth is significantly lower than a local area network. Thus cloud backup and restoration is much slower and costs more than a traditional on-site backup [6].

The cryptographic hash algorithm used in above described paper use MD5, SHA-1 cryptographic hash function. In MD5 the hash collision occurs [13], also in SHA-1 attacks are possible and thus the secrecy of the data cannot be maintained.

Thus the paper focus on maintaining the secrecy of the data, good balance between cloud storage capacity and deduplication time saving i.e. for storing the personal documents into the cloud the redundant files should not be copied and the space and time should not be wasted.

3. PROBLEM STATEMENT

The backing up of redundant files and data increases the backup window size this results in over utilization of Network resources and require too much additional storage capacity to hold unnecessary backup data. Also the security issue arises when the storage resources are controlled by third party.

Thus the problem that focused here is maintaining proper backup window size and maintaining the secrecy of the data to be stored. So the backup process may be performed securely and efficiently over the network.

4. IMPLEMENTATION DETAILS

A. System Architecture

The system is comprises of two parts client side and cloud side. The client components perform data deduplication at the local level. The client machine generates the fingerprint index of the data chunks. When client uploads the data to the cloud, the compressed chunks of the data are sent to the cloud storage. At the cloud side the fingerprints of the received index are stored in the Global index. The Figure.1.Data Deduplication Architecture depicts the components in the systems. The components of the system are described as follow:

File Size Filter

Large number of tiny files present in the client machines or even in the system of any normal user. These tiny files are of very small size say less than 10 KB. To reduce the metadata overhead the system first filter out such tiny files in the file size filter before performing the deduplication process. These tiny files are directly sent to data deduplicator and hashing unit. The non tiny files are further sent to the Data chunking.

Data Chunking Unit

The data chunking unit splits the large file in small data chunks. The non tiny files received by the data chunking from the file size filter are split in to smaller data chunks. The maximum chunk size we have considered here is of 256kb. To maintain the fix size of chunks static chunking has been performed on the large files.

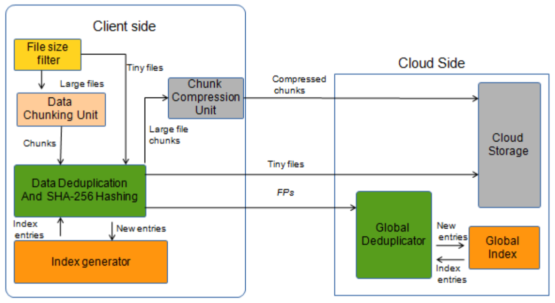

Figure.1.Data Deduplication Architecture

Data Deduplication and SHA-256 hashing unit

After performing data chunking process in data chunking unit. The data deduplication and SHA-256 hashing unit deduplicates the data and generates fingerprint of the data. Here SHA-256 hashing is performed on the data chunks to generate the fingerprint. As SHA-256 hash function is the collision-resistant hash function. The message digest size of fingerprint produce by SHA-256 is 256 bit [12],[14]. This SHA-256 generated fingerprint is employed to detect duplicate chunks in the index. To achieve data deduplication efficiency duplicate fingerprints are first detected in the local index. If fingerprint matches with the new fingerprint then the data is considered as the duplicate data and it is not uploaded to the cloud. If the fingerprint does not match with the existing fingerprint in the local index then it is considered as the new fingerprint. Further chunks of non-tiny files are sent to compression unit and tiny files are directly uploaded to the cloud storage after deduplicating them. Thus only the unique data chunks and the fingerprints are sent to cloud storage.

Chunk Compression unit

The Chunk Compression unit implements DEFLATE compression algorithm [11] to compress the chunks of larger files. The deduplicated chunks of data are compressed and uploaded to the cloud storage. Compressing the chunks reduces the size of the data to be sent. Thus the bandwidth occupied by the compressed chunks is less as compared to uncompressed chunks to be transferred over the WAN. So the bandwidth space utilization is reduces by transferring compressed chunks over the network.

Index generator

According to the file information the incoming file chunks are directed to the chunk index each entry to the index contains fingerprint (fp) of the chunk, length of the chunk (len) and the container id (cid). The fingerprint works as the key element to identify the similar data chunks in the index. The SQL index is used to store the details of the data chunks and to identify similar data chunks by their fingerprints. As the SQL index can perform faster traversing through the index, the fingerprints can be found more efficiently in the index. Also large number of fingerprints can be stored persistently in SQL index. Thus the computational overhead of matching of fingerprint for every session is reduced.

The Index is present at the client side and the cloud side. The index at the client side is the local index containing the details of the local chunks deduplicated at the client side. The global index consists of the fingerprints of the chunks at the cloud side, as well as the fingerprints of the chunks that are uploaded by the client side. Before uploading the chunk from the client side fingerprints of the chunks are first compare at the local side. If the fingerprint is present at local index then the data chunks and their fingerprints are not transferred to cloud. Thus the data chunk with similar fingerprint is considered as the duplicate data chunk and so it is not uploaded to the cloud. The data chunks whose fingerprints are not found in the global index chunks are considered as the new chunks and their fingerprint is stored as the new entry to the global index. This technique prevents the duplicate data chunks to be transferred from client side to the cloud side. Thus the bandwidth is properly utilized by not sending the duplicate chunks.

B. Mathematical Model

Let A be the data deduplication system where data sets are deduplicated to upload them to the cloud.

$A=\{Ds, C, Fp, I, Co, U\}$

Ds represent initial state, where data set are given as input.

$Ds = \{Ds_1, Ds_2, \ldots, Ds_n\}$

The data chunking process is represented by C. The chunks C of the data are formed in this process.

$C=\{C_1, C_2,\ldots, C_n\}$

Fingerprint generation phase:

SHA-2 hash function from family of cryptographic hash functions is implemented on the data chunks C.

$H:K\times C\rightarrow Fp$ Here Fp represents the fingerprint. K represents key. C represents data chunks on which SHA-2 is implemented. So the set of fingerprint can be represented as.

$Fp = \{Fp_1, Fp_2, \ldots, Fp_n\}$

The duplicate chunks are identified on the basis of the fingerprint. The SQL Index is represented as

$I= \{i_1, i_2, \ldots i_n\}$

Co represents the compressed data to be sent over the network.

$Co= \{Co_1, Co_2, \ldots Co_n\}$

U represents the data uploading phase. Here only unique and compressed data chunks are sent over the network.

C. Algorithm

In this section the algorithmic description of the whole system can be done as follows.

Step 1: Different types of existing files are given as the input.

Step 2: Files are classified on the basis of there size.

```
If (file_size < 10KB)
        Store file in segment store
else
        Send file to data chunking
```

Step 3: Split the files in to the chunks of 1MB each.

Step 4: Apply SHA-256 on 250 KB size chunks of data to generate 256 bit hash fingerprint, to make the data chunk secure and improve collision resistance.

Step 5: Identify the duplicate chunks and store the new chunk fingerprint, chunk size of file extension in the SQL index.

```
If (existing fingerprint = new fingerprint)
        do not store the fingerprint it is the duplicate data
    else
```

store it as new fingerprint

Step 6: Compress the 250 KB chunks before uploading them.

5. Evaluation

A. Data set

The dataset considered for evaluation consist of existing files in authors Personal Computer. Also the personal laptops and research desktop were considered for Data set. TABLE 1. shows that files with different type of extensions used as a data set.

B. Results

Reduced Chunk Size

After performing chunks of different files. The size of chunks were compared with the size of compressed chunks. It was observed that the compressed chunk size consumes less bandwidth as compared to uncompressed chunks. TABLE I. Shows the observed chunks size. The max chunk size considered is 256 kb. Also as per the file content the size of compressed chunk varies. Thus the compressed chunk size range from 2 KB to 248 KB. Fig.2. shows the reduction in the chunks size after performing the compression. Here the effect of compression on the data chunks can be observed easily from the graph.

TABLE I. Observed Compressed chunk size

File Type	Size	Uncompressed chunks size	Compressed Chunk Size
.mp3	11 MB	56 KB to 250 KB	17 KB to 248 KB
.txt	2.69 MB	8 KB to 250 KB	2 KB to 3 KB
.pptx	1.30 MB	86 KB to 250 KB	75 KB to 225 KB
.pdf	1.66 MB	203 KB to 250 KB	120 KB to 227 KB

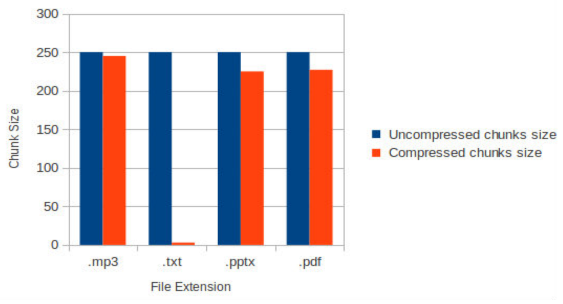

Figure.2.Chunk size comparison

As observed in Figure.2. The compression is more effective on files in the ".txt" format, as compared to rest of the three file formats shown in the graph. Similarly ".pptx " and ".pdf" file formats require less bandwidth as compared to ".mp3" file format.

Backup Window

The backup window size mainly depends on the volume of the transferred data set and the available network bandwidth. The Compressed data sent from the chunk compression unit reduces the volume of data sent on the WAN. It reduces the total chunk size as 250 KB segments are compressed and sent further to the cloud storage. The compressed size of 250 KB chunk is reduced as per the content of the data chunk. Thus the reduction in backup window size can be observed.

6. Discussion

By implementing the birthday attack technique hash collision was possible in case of MD 5 and SHA 1[]. as the message digest size of SHA 256 is more than SHA1 and MD 5, SHA 256 is more secure. Thus by implementing the SHA 256 on the data chunks increases the secrecy of the data chunks in case of third party storage.

By saving the fingerprint in the SQL index allows to persistently store the fingerprints of the data chunk. Thus it makes easy to search for the fingerprints of the data chunks that are already uploaded by the user.

Compressing the data chunk reduces the bandwidth requirement of the data chunks thus the bandwidth can be properly utilized. So the data chunks can easily be transmitted in the low bandwidth network.

7. Conclusion and Future Scope

The block level deduplication approach along with data chunk compression used in this paper reduces the bandwidth utilization due to its fix size chunks and compression. Thus by sending the compressed data over the network the bandwidth can be wisely used for data transfer. By applying the SHA-256 cryptographic hash function the security of the data chunks is improved and also the hash collision is reduced. The implementation of deduplication technique at client side and cloud side reduces the redundant data to large extent. Thus the availability of data and proper utilization of storage space can be managed with the Deduplication.

In this paper sending the deduplicated data from client side may increase the chance of retransmitting the same type of data that may be present at the cloud side though it s not sent by the same client / user. Thus sending only the unique data and implementing faster fingerprint searching techniques are the important objective for the future scope.

References

[1] Yinjin Fu, Hong Jiang, Senior Member, IEEE, Nong Xiao, Member, IEEE, Lei Tian, Fang Liu, and Lei Xu , "Application-Aware Local-Global Source Deduplication for Cloud Backup Services of Personal Storage" , IEEE Transaction on parallel and distributed systems, vol. 25, no. 5, May 2014.

[2] Y. Tan, H. Jiang, D. Feng, L. Tian, Z. Yan, and G. Zhou, ''SAM: A Semantic-Aware Multi-Tiered Source De-Duplication Framework for Cloud Backup,'' in Proc. 39th ICPP, 2010, pp. 614-623.

[3] Y. Fu, H. Jiang, N. Xiao, L. Tian, and F. Liu, ''AA-Dedupe: An Application-Aware Source Deduplication Approach for Cloud Backup Services in the Personal Computing Environment,'' in Proc. 13th IEEE Int'l Conf. CLUSTER Comput., pp. 112-120, 2011.

[4] Amrita Upadhyay, Pratibha R Balihalli, Shashibhushan Ivaturi and Shrisha Rao, "Deduplication and Compression Techniques in Cloud Design", 978-1-4673-0750-5/12/$31.00 ©2012 IEEE.

[5] P. Anderson and L. Zhang, "Fast and Secure Laptop Backups with Encrypted De-duplication," in Proceedings of the 24th international conference on Large Installation System Administration (LISA'10), pp. 29-40, 2010.

[6] P.Neelaveni and M.Vijayalakshmi, "A Survey on Deduplication in cloud storage", Asian Journal of Information Technology 19(6): 320-330,2014.

[7] D. Meister, "Advanced Data Deduplication Techniques and their Application" , 2013.

[8] M. Vrable, S. Savage, and G. M. Voelker, " Cumulus: File system Backup to the Cloud," in FAST'09, Feb. 2009.

[9] C. Liu, Y. Lu, C. Shi, G. Lu, D. Du, and D.-S. Wang, ''ADMAD: Application-Driven Metadata Aware De-Deduplication Archival Storage Systems,'' in Proc. 5th IEEE Int'l Workshop SNAPI I/Os, pp. 29-35, 2008.

[10] www.EMC.com, "An EMC Perspective on Data De-Duplication for Backup".

[11] http://www.zlib.net/feldspar.html

[12] https://www.tbs-certificates.co.uk/FAQ/en/475.html

[13] http://en.wikipedia.org/wiki/SHA-1

[14] Kiatchumpol Suttisirikul, Putchong Uthayopas," Accelerrating the cloud backup using GPU based data deduplication", 2012 IEEE 18th International conference on parallel and distributed systems, 2012.