Effective Deduplication and selective Compression in AWS S3

Praveen Kumar Chukkapalli
Department of Computer Science
University of Houston
Houston,Tx
pchukkap@cougarnet.uh.edu

Harikrishna Vinjam
Department of Computer Science
University of Houston
Houston,Tx
hvinjam@cougarnet.uh.edu

Venkata Siva Sai Vundela
Department of Computer Science
University of Houston
Houston,Tx
vvundela@cougarnet.uh.edu

Abstract—Cloud Storage has become a popular pay to go service to store data in an easily accessible way. This also leverages the concerns of operational cost management, and every user strives to optimize their utilization. Our main goal is to make cloud storage more efficient and reduce operating costs of the cloud environment. The S3 Cloud storage service is improvised by Deploying effective De-duplication and selective Compression of files to improve the storage utilization thereby optimizing the operational costs to the end user.

Index Terms—Compression, Deduplication, E-tag, Hash, Lambda.

1 Introduction

The significance of cloud computing on industries and end users cannot be overstated: the ubiquitous presence of software that runs on cloud networks has altered many elements of daily life. Startups and enterprises can reduce costs and expand their offerings by embracing cloud computing rather than purchasing and managing their own hardware and software. Global accessible apps and internet enterprises can be now created by Individual developers. Researchers may now share and analyze data on previously unimaginable scales. Furthermore, internet users can easily access software and storage in order to produce, share, and store digital media in numbers that much exceed the computing power of their personal devices.

Despite the growing presence of cloud computing, There are ongoing concerns on the optimization of operational costs. In order to improve the S3 Cloud storage service by providing optimal operational costs, Effective Deduplication and compression techniques have been proposed.

1.1 Methodologies

1.1.1 Deduplication Approach

- Uploading the same object multiple times in an S3 bucket will occupy extra space which in turn will increase the costs for storage.
- Whenever a file is uploaded to bucket, its hash value is calculated and iterated over the list of hash values of existing files stored in Dynamo DB.
- If Hash value is matched, the file is categorized as duplicated and removed from the S3 bucket saving the cloud storage resources.

1.1.2 Selective compression with analysis.

- In Selective Compression, Last Used date of the object of file will be tracked along with their ETag value in the DynamoDB.
- If the File isn't being used Frequently i.e. if the last used date is beyond the set threshold value, the file will be compressed and stored in S3.
- prevents the additional storage utilization by files and the analysis prevents excess operational costs of compression and decompression.

2 Literature review

There are few works related to effective cloud storage management techniques as represented in [1]. However, there were very few works related to de-duplication of files in AWS Cloud storage services. Author in [3] proposed a Cloud based file deduplication system that works by similarity detection with variable-size chunking method. However, this proved effective only on ASCII and binary files and this motivated us to use the E-Tag unique identification parameter for files while in AWS or to calculate the Hash Values of file by checksum in other environments for data deduplication.. Author in [3] also proposed the concept of adaptive deduplication techniques for mobile devices improving the operational efficiency.

Selective decompression is a scenario based experimental approach to compress the files in the cloud storage which are not being used for a long time in an organization thereby reducing the operational and maintenance costs. There are no existing works related particularly to this domain, but the author in [5] leveraged that only deduplication is not effective enough to manage the storage effectively. This motivated us to include selective compression along with deduplication to improve the overall cloud storage efficiency.

3 Implementation

First a S3 bucket named "myefficientbucket" was created and made it public so that files can be uploaded to it via a web page. Next a web page is created to upload files to s3 bucket. This web page is created using the "Flask" framework and was hosted in an EC2 instance. Next a DynamDB table called "checkSumTable" was created to store checksum of the files uploaded to s3 bucket. This checksum is used to identify whether the uploaded file to the s3 bucket is a duplicate or not. This DynamDb table has the following attributes:

- checkSum: checksum of the file
- fileName : name of the file
- Compressed: whether the file is compressed or not if it is compressed it contains "yes" value otherwise "no"
- compressedSize(CS): size of the file after compression
- originalSize(OS): size of the original size
- compressionRatio(CR): Compressed ratio of file compared to the orginal file.

$$CR = (OS - CS)/OS$$

- lambdaRuntime: time taken to compress the file
- memorySaved: originalSize - compressedSize
- uploadedTime: uploaded date and time of the file. This attribute will be used to compress the file.

A lambda function "my-s3-function" is created to handle deduplication. This lambda function was given permission to delete files from S3 bucket and also to access checksum from DynamoDB table. Whenever a file is uploaded to the S3 bucket, this function will be triggered and the checksum of that file is calculated. If that checksum is already present in the dynamodb table it means it is a duplicate file and will be deleted from the S3 bucket otherwise this new checksum is added to the dynamodb table and the file is allowed to stay in the S3 bucket. Next to implement selective compression a new lambda function "scheduledFunction" is created. For this lambda, permissions are given to get

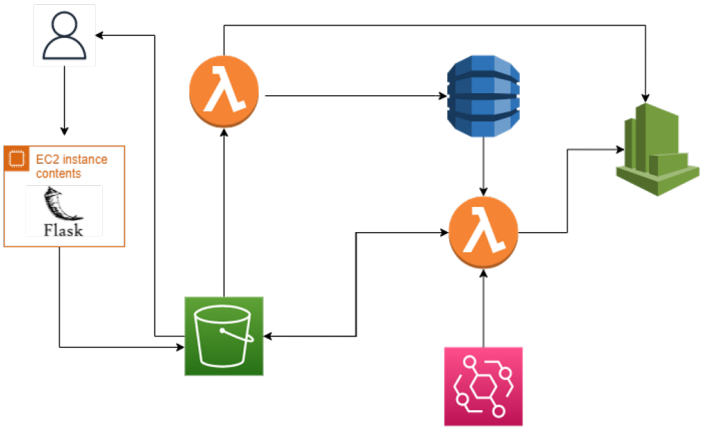

Fig. 1. Cloud Architecture Diagram

and put objects in S3 bucket and also permission to access DynamoDB table. This lambda function is triggered periodically by Eventbridge service. Upon every trigger this function gets data from DynamoDb table to check how long the files have not been used in S3 bucket. If a file hasn't been used for more than 30 days that file will be compressed and will be stored in an archive folder. And also it will update file information in DynamoDB table like compressionRatio, memorySaved, changing compressed attribute to "Yes" etc. Next in Event Bridge service a schedule is created to trigger "scheduledFunction" periodically.

4 Results

Assumed that total 9 types of file are uploaded to S3 of similar size periodically. Operational costs for AWS services are calculated based on the AWS pricing rate for the model, we currently operating on.

Month	Original Operational cost	Reduced operational cost	lambda cost	Net total cost saving
1	0.01161068173	0.005401390176	0.003037581453	0.003171710099
2	0.02322136346	0.01080278035	0.003037581453	0.00938100165
3	0.03483204519	0.01620417053	0.003037581453	0.01255271175
4	0.04644272692	0.0216055607	0.003037581453	0.01572442185

Fig. 2. overall metrics of cloud storage service with mentioned methodologies in action.

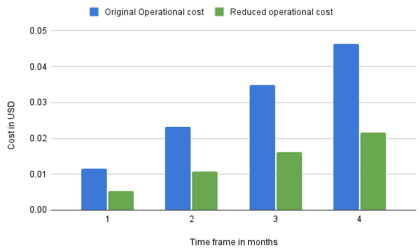

Fig. 3. Visualization of Data from Fig.1

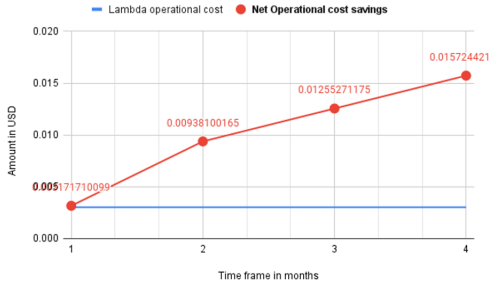

Fig. 4. caption need to be changed

4.1 Analysis

Fig 2 consists of the numerical results calculated on basis of current AWS pricing structure. Original operational costs depicts the cost of storing the data files in original format. Reduced operational costs depicts the cost of storing the data files after applying the mentioned deduplication and selective compression algorithms. lambda costs refers to the operational costs of running lambda function to perform mentioned operations on the data files. The final savings

of operational costs after deducting lambda cost is the net savings achieved, which is depicted in last column if the table.

Fig 3 represents the visual comparison of original operational cost and reduced operational cost over the time period in months. It can be observed that over the period, forecasted original operational cost keeps on increasing as its assumed that user uploads similar content every month to the cloud storage. similarly, the reduced operational costs keeps increasing along with time. But the difference between original and reduced costs also keeps on increasing, which depicts that the net total savings is increasing every month.

Fig 4 also depicts the same in line chart visualization. It can be observed that the total net savings is forecasted to increase in non linear fashion tough the similar kinds of files are uploaded each month. This concludes that results depicts the positive impact on efficient cloud storage following the mentioned approaches.

5 Conclusion

We are able to improve the operational cost savings in typical assumed scenarios. To work on the decompression techniques in runtime when an user requests for the compressed file. To improve it further into a scalable solution also improving the run time operational costs for compression and decompression.

References

[1] Ravneet Kaur, Inderveer Chana, and Jhilik Bhattacharya. Data deduplication techniques for efficient cloud storage management: a systematic review. *The Journal of Supercomputing*, 74(5):2035–2085, 2018.

[2] Rodel Miguel, Khin Mi Mi Aung, et al. Hedup: Secure deduplication with homomorphic encryption. In *2015 IEEE International Conference on Networking, Architecture and Storage (NAS)*, pages 215–223. IEEE, 2015.

[3] Le-Juan Tan, Wen-Bin Yao, Zheng-Yang Liu, and Yi-Xian Yang. Cdfs: a cloud-based deduplication filesystem. *Advanced Science Letters*, 9(1):855–860, 2012.

[4] Ryan NS Widodo, Hyotaek Lim, and Mohammed Atiquzzaman. Sdm: Smart deduplication for mobile cloud storage. *Future Generation Computer Systems*, 70:64–73, 2017.

[5] Zheng Yan, Lifang Zhang, DING Wenxiu, and Qinghua Zheng. Heterogeneous data storage management with deduplication in cloud computing. *IEEE Transactions on Big Data*, 5(3):393–407, 2017.