Volume 5 Issue V, May 2017
ISSN: 2321-9653
International Journal for Research in Applied Science & Engineering Technology (IJRASET)

Optimization of Backup Storage by Reducing Fragmentation in Distributed Environment

Sandeep Wagh[1], Prof. Sagar Bhakre[2]
[1,2] Computer Science & Engineering, Ballarpur Institute of Technology

Abstract: ***In modern backup systems, Deduplication plays a vital role in the elimination of duplicate data in a storage system which one of the technique to reduce storage costs. Deduplication divides a backup stream into variable sized chunks of data used to map stored chunks to their physical addresses. These chunks of data are physically distributed and create a problem known as fragmentation. Primarily fragmentation categorized into sparse and out-of-order containers. The sparse container adversely affects the performance while restoring the database and garbage collection effectively, while the out-of-order container brings an adverse effect on the performance issue if the restore cache built is small. To minimize the fragmentation problem, we implement the History-Aware Rewriting algorithm (HAR) and Cache-Aware Filter (CAF). HAR will collect the historical information in backup systems to identify and reduce sparse containers, and CAF to restore cache knowledge to find the out-of-order containers that impacts restore performance. We exploit Container-Marker Algorithm (CMA) to gather valid containers instead of valid chunks which help in garbage collection. My results help to prove how HAR, CMA improves the restore performance.***
Keywords: Deduplication, Fragmentation problem, HAR, CMA.

I. INTRODUCTION

The existing challenge in the backup storage system is the management we need to handle the ever-increasing volume of data. To face challenges and to convert scalable output, deduplication technique is very well known and new techniques and research are being done to make this technique more optimized. Data deduplication is one of the special techniques used in data compression. These work by eliminating the duplicate copy in the storage system. Hence it helps in improving the total experience and utilization of the data storage in the dataset which we provide as input. Also help in managing the data transfer via network, a cloud where the amount of data transfer will get reduced significantly. In deduplication, we keep only one copy of data and eliminating redundant data and refer other data which are redundant to the same copy original copy. Deduplication can happen either in the file level system or can happen at the block level. The fragmentation is classified into two categories: sparse containers and out-of-order The former reduces restore performance, which might be self-addressed by increasing the size of the cache used for restoration. HAR considerably improves restore performance with a small decrease of deduplication ratio. We have a tendency to develop CAF to take advantage of cache information to spot the out-of-order containers that might hurt restore performance. CAF is employed within the hybrid theme to boost restore performance underneath restricted restore cache while not a major decrease of deduplication magnitude relation. So as to scale back the data overhead of the garbage collection we have a tendency to propose CMA that identifies valid containers rather than valid chunks within the garbage collection. Deduplication also can also happen at the block level, thus eliminate a duplicate set of blocks of knowledge that occur in non-identical files.Although the data deduplication methods bring lots of advantages to the user along with security and privacy issues. Convergent encoding has been planned to enforce dataset confidentiality whereas creating deduplication possibleAnalyzing the visual content may not be sufficient to capture users' privacy preferences. T-tags and other metadata are indicative of the social context of the image, including where it was taken and why [4], and also provide a synthetic description of images, complementing the information obtained from the visual content analysis.

II. LITERATURE SURVEY

Deduplication technologies are more and more being deployed to scale back value and increase space efficiency in company knowledge centres. However, previous analysis has not applied deduplication techniques to the path which is requested for latency sensitive, primary workloads.

This is often primarily thanks to the additional latency these techniques were introduced. Inherently, deduplicating data that is present on the disk causes fragmentation [2] that may increase seek for sequent successive reads of a similar data thus, increasing latency. Additionally, deduplicating knowledge needs additional disk IOs to access on-disk deduplication information. During this

paper, we have a tendency to propose AN inline deduplication resolution, iDedup [1], for primary workloads, whereas minimizing additional IOs and seek. Our algorithmic rule relies on 2 key insights from real-world workloads: i) abstraction vicinity exists in duplicated primary knowledge, and ii) temporal vicinity exists within the access patterns of duplicated data. Mistreatment the primary insight, we have a tendency by selection deduplicated solely sequences of disk blocks. This reduces fragmentation and seeks caused by deduplication. The second insight permits U.S. to switch the high-priced, on-disk, deduplication information with a smaller, in-memory cache. These techniques alter U.S. to exchange capability savings for performance, as incontestable in our analysis with real-world workloads.

Data deduplication has recently become a very well known thing in most auxiliary storage and even in some primary storage for the capability improvement purpose. Other than its write performance, browse performance of the deduplication storage has been gaining quite an importance with a good vary of its deployments. During this paper, we have a tendency to emphasize the importance of browsing in the performance in reconstituting a knowledge stream from its distinctive and shared chunks physically spread over deduplication storage. We have a tendency to freshly introduce a browse performance indicator referred to as Chunk Fragmentation Level (CFL) [3].

We have a tendency to conjointly validate that the CFL is incredibly effective to point browse performance of the deduplication storage by understanding theoretical performance model and intensive experiments.

Backup storage systems typically take away redundancy across backups via inline deduplication that works by referring duplicate chunks of the newest backup to those of existing backups. Inline deduplication reduces restore performance of the newest backup owing to fragmentation and complicates deletion of terminated backups owing to the sharing of knowledge chunks. Whereas out-of-line deduplication addresses the issues by forward pointing existing duplicate chunks to those of the newest backup, it introduces extra I/Os of writing and removing duplicate chunks. We have a tendency to style and implement RevDedup[4], associate degree economical hybrid inline present in data set and outof-line deduplication system for backup storage. Itapplies for the coarse-grained inline deduplicationdata to delete the duplicates of the newest backup, and so fine-grained out-of-line reverse deduplication to delete the duplicates from older backups. Our reverse deduplication style limits the I/O overheadand prepares for economical deletion of terminated backups.

Data deduplication database finds and exploit redundancy of data between totally different information blocks. The foremost common approach divides information into chunks and identifies redundancies via fingerprints. The file content is remodeled by combining the chunk fingerprints that square measure hold on consecutive during a file formula. The corresponding file formula information will occupy a major fraction of the full disc space, particularly if the deduplication magnitude relation is incredibly high. We have a tendency to propose amix of economical and scalable compression schemes to shrink the file[6][7]. At simulation shows that these ways will compress file recipes by up to ninety three.

III. PROPOSED METHODOLOGY

Trace back (IPT) is a method that enables the proper identification of the source of a packet on a network and may at the same time provide the full or partial path reconstruction of that packet as it traverses the network. Proposed mechanism is consists for following steps.

A. Convergent encryption

Convergent secret writing provides information confidentiality in deduplication. A user tries to derive a convergent key from every original information copy and encrypts the data copy with the help of convergent key. Additionally, the user conjointly tag for the data copy, specified the tag are accustomed find duplicates.

Here, we try to assume that the correctness of the tag holds property, i.e., if 2 information copies area unit constant, then their tags area unit constant.

To find repeating copies, the user initial sends the tag to the server aspect to envision if the identical copy has been already keep. Note that each the convergent key and therefore the tag area unit severally derived and therefore the tag cannot be accustomed deduce the convergent key and compromise information confidentiality.

B. HAR Architecture

We have pool of container which can provide a storage service on disk. We can use fingerprint indexes for this. The disk is useful for keeping the finger print index, and hot part which is in memory. For writing the chunk we use container buffer which is present inside memory the dataset are assigned with the unique ID, say DS1.The transaction orhistorical data we have are stored on disk which consists of ID for e.g. DS1the transactional or historical data are divided into 3 main parts: Sparse container of HAR for inherited IDs, the optimal replacement cache and CMA for container manifest.

C. History-Aware Rewriting Algorithm

1) Whenever the backup starts, HAR holds the IDs of sparse container which are inherited to build inmemory S inherited structure. Whenever backup start, HAR rewrites all duplicate chunks whose container IDs exist in S inherited.
2) Along with that HAR also maintains a structure in built knows as S emerging to monitor or housekeeping the container referred by backup and maintains their utilization factor.
3) S emerging is used to record the utilization factor and each dataset or record consist of utilization factor of each container
4) Once the backup is done HAR uses the technique to get the record of higher utilization from S emerging. S emerging has the list of all emerging container which are sparse.
5) S emerging can be directly sent to disk as the size of S emerging is small due to our second observation
6) Hence from the observation we can come to know that if the value of chunks is more than they are rewritten in next backup. It would hamper the performance and cause bottleneck issues.
7) To mitigate this effect HAR set the limit as 5% for rewriting, tis would avoid too much rewrite on future backup. HAR makes use of limit defined for rewriting for segregating too many sparse containers in S emerging
8) HAR helps in estimating the rewrite ratio for the coming backup. Specifically it calculates the size for the chunks which are rewritten for each emerging sparse container.
9) This is done using the help of the utilization factor that is calculated from container size. The rewrite ratio is later calculated as the total of all estimated size dividing by the current backup size, which maybe give us approx.The value of rewrite ratio for the coming backup or next backup.

D. Optimal Restore Cache

To reduce the side effects of out-of-order containers on restore performance, we tend to implement Belady's best replacement cache. Implementing the optimal cache (OPT) has to apprehend the long run access pattern. We will collect such data throughout the backup, since the sequence of reading chunks throughout the restore is simply constant because the sequence of writing them throughout a backup. Once a chunk is processed through either elimination or over-writing its Container ID, its Container ID is understood. We tend to add access record in the information that is collected .Every access record will solely hold a Container ID. Consecutive accesses to the identical container will be incorporated into are the cord. This part of historical data will be updated to disks sporadically, and so wouldn't consume a lot of memory The entire sequence of access records will consume hefty memory once out-of-order containers are dominant. Forward every container is accessed fifty times intermittently and also the average utilization is five hundredth, the entire sequence of access records of a one TB stream consumes over a hundred MB of memory.

Rather than checking the entire sequence of access records, we can take a help of slide window to look into a fixed-sized a part of the long run sequence, as a near optimal theme. The memory foot-print of this near optimal theme is thus delimited.

E. Container-Marker Algorithm

Existing garbage pickup schemes have faith in merging thin containers to reclaim invalid chunks within the containers. Before merging, they need to spot invalid chunks to work out utilizations of containers, i.e., reference management. Existing reference management approaches area unit inevitably cumbersome owing to the existence of huge amounts of chunks. HAR

Volume 5 Issue V, May 2017
ISSN: 2321-9653
International Journal for Research in Applied Science & Engineering Technology (IJRASET)

naturally accelerates expirations of thin containers and therefore the merging is not any longer necessary. Hence, we'd like to not calculate the precise utilization of every container we tend to copy the Container-Marker algorithmic program (CMA) to with efficiency confirm that containers area unit invalid. CMA assumes users delete backups during a FIFO theme, during which oldest backups area unit deleted initial.

IV. SYSTEM ARCHITECTURE

Main purpose of our model is to provide Security to Authorized Deduplication in daily backup and helping it to reduce the fragmentation via exploiting backup history and cache knowledge. Deduplication comes with many issues like security, privacy of user. The security measures taken in deduplication are not good enough. Specially the old measures to protect the data against encryption. The earlier method encrypt with the own key making it impossible to access. Convergent encryption used widely for making deduplication possible. Hence hash value is used to protect the data copy. After the generation of key the user keeps the key and sends the cipher text to backup system. As the key is derived from the main data the duplicate
will have the same data which would prove out to be easier to access the data in feasible way. A new protocol is introduced for user data to identify the duplicate data is the copy of main file hence need not upload the same in the server. A convergent key can be used to decrypt the data which is encrypted in the server. This key can be downloaded from the server itself.This encryption can help to prevent the unauthorized access of data in the server for the user.

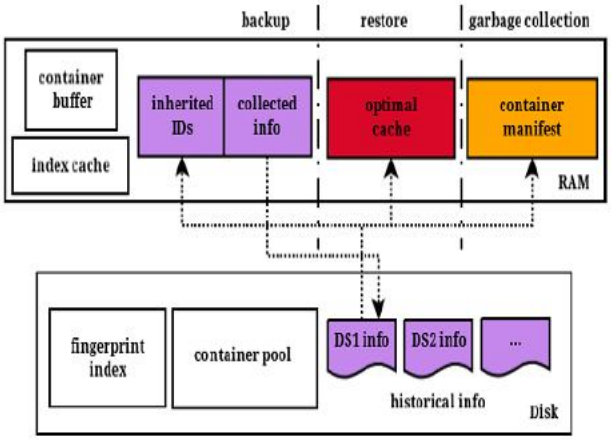

Fig. 1 System Architecture

V. CONCLUSIONS

Fragmentation one of the main cause which considerably decreases restore and garbage collection efficiencies in deduplication backup systems. Sparse basically tell us about the maximum restore and out-of-order tells us about restore performance. HAR help us to identify and rewrite sparse with the knowledge of history. We also came to the conclusion that an optimal caching scheme which is optimal and hybrid algorithm act as a complementaryto HAR for reducing the impact of out-of-order case. HAR and OPT helps to optimize the restore performance in deduplication ratio. HAR helps to optimize both deduplication ratio and restore performance. As to reduce deduplication in hybrid scheme we involved CAF to reduce deduplication ratio in hybrid scheme. We can adapt CAF for optimizing the rewriting algorithms. Container-Marker Algorithm (CMA) is introduced to identify valid containers instead of valid chunks. CMA is bounded by the number of containers; it is more costeffective than the number of chunks.

REFERENCES

[1] K. Srinivasan, T. Bisson, G. Goodson, and K. Voruganti, “iDedup:Latency-aware, inline data deduplication for primary storage,” in Proc. USENIX FAST, 2012. [1]

Volume 5 Issue V, May 2017
ISSN: 2321-9653

[2] Y. Nam, G. Lu, N. Park, W. Xiao, and D. H. Du, "Chunk fragmentation level: An effective indicator for read performance degradation in deduplication storage," in Proc. IEEE HPCC, 2011. [2]
[3] F. Guo and P. Efstathopoulos, "Building a highperformance deduplication system," in Proc. USENIX ATC, 2011. [3]
[4] J. Wei, H. Jiang, K. Zhou, and D. Feng, "MAD2: A scalable highthroughput exact deduplication approach for network backup services," in Proc. IEEE MSST, 2010 [4]
[5] M. Kaczmarczyk, M. Barczynski, W. Kilian, and C. Dubnicki [5]
[6] M. Lillibridge, K. Eshghi, D. Bhagwat, V. Deolalikar, G. Trezise, and P. Camble, "Sparse indexing: large scale, inline deduplication using sampling and locality," in Proc. USENIX FAST,2009. [6]
[7] D. Meister and A. Brinkmann, "dedupv1: Improving deduplication throughput using solid state drives (SSD)," in Proc.IEEE MSST, 2010. "Reducing impact of data fragmentation caused by inline deduplication,"in Proc. ACM SYSTOR, 2012. [7]