www.ijecs.in
International Journal Of Engineering And Computer Science ISSN: 2319-7242
Volume 5 Issues 6 June 2016, Page No. 16824-16829

Protection Preserving Public Supportable System In Cluster Based Distributed Storage

Nivethaa Varshinie.R[1], Rajarajan.A[2]

[1]M.E. Student, Department of CSE, Parisutham Institute of Technology and Science, Tamil Nadu, India [2]Asst. Professor, Department of CSE, Parisutham Institute of Technology and Science, Tamil Nadu, India

[1]varshinie05@gmail.com, [2]rajarajan.ag@gmail.com

***Abstract*— to secure outsourced information in distributed storage against defilements, adding adaptation to internal failure to distributed storage together with information uprightness checking and disappointment reparation gets to be basic. As of late, recovering codes have picked up ubiquity because of their lower repair transfer speed while giving adaptation to non-critical failure. Existing remote checking systems for recovering coded information just give private examining, requiring information proprietors to dependably stay online and handle reviewing, and additionally repairing, which is some of the time unreasonable and also all the distributed data are stored in same functional location ,so search and data retrieval takes much time. This time delay will affect the distributed storage efficiency. In this project an open examining plan for the recovering code based distributed storage is proposed and also Attribute Based Clustering Technique (ABCT) For Distributed Data. The ABCT will recover the issue of time delayness and makes the system more efficient. We randomize the encode coefficients with a pseudorandom capacity to protect information security. The ABCT achieves the much faster performance data searching and retrieval. Broad security examination demonstrates that our plan is provable secure under arbitrary prophet model and trial assessment shows that our plan is exceptionally productive and can be attainably coordinated into the recovering code based distributed storage**.

Keywords: **Clustering, ABCT, Distributed System.**

1 INTRODUCTION

Nowadays, the ever-growing volume and value of digital information have raised a critical and increasing requirement for data protection in the personal computing environment. Cloud backup service has become a cost-effective choice for data protection of personal computing devices [1], since the centralized cloud management has created an efficiency and cost inflection point, and offers simple offsite storage for disaster recovery, which is always a critical concern for data backup. And the efficiency of IT resources in the cloud can be further improved due to the high data redundancy in backup dataset [2]. Data deduplication, an effective data compression approach that exploits data redundancy, partitions large data objects into smaller parts called chunks, represents these chunks by their fingerprints (i.e., generally a cryptographic hash of the chunk data), replaces the duplicate Clustering: Cluster analysis or clustering is the task of grouping a set of objects in such a way that objects in the same group (called a cluster) are more similar (in some sense or another) to each other than to those in other groups (clusters).Cluster analysis itself is not one specific algorithm, but the general task to be solved. It can be achieved by various algorithms that differ significantly in their notion of what constitutes a cluster and how to efficiently find them. Popular notions of clusters include groups with small distances among the cluster members, dense areas of the data space, intervals or particular statistical distributions.

Cluster analysis as such is not an automatic task, but an iterative process of knowledge discovery or interactive multi-objective optimization that involves trial and failure. It will often be necessary to modify data preprocessing and model parameters until the result achieves the desired properties. Clustering servers is completely a scalable solution. You can add resources to the cluster afterwards. If a server in the cluster needs any maintenance, you can do it by stopping it while handing the load over to other servers. Among high availability options, clustering takes a special place since it is reliable and easy to configure. In case of a server is having a problem providing the services furthermore, other servers in the cluster can take the load the cloud side before data transfer over WAN. The former only eliminates intra-client redundancy with low duplicate elimination ratio by low-latency client-side duplicate data check, while the latter can suppress both intra-client and inter-client redundancy with high deduplication effectiveness by performing high-latency duplication detection on the cloud side. Inspired by Cloud4Home [9] that enhances data services by combining limited local resources with low latency and powerful Internet resources with high latency, local-global source deduplication scheme that eliminates intra-client redundancy at client before suppression inter-client redundancy in the cloud, can potentially improve deduplication efficiency in cloud backup

services to save as much cloud storage space as the global method but at as low latency as the local mechanism.ALG-Dedupe, an Applicationaware Local-Global source deduplication scheme that not only exploits application awareness, but also combines local and global duplication detection, to achieve high deduplication efficiency by reducing the deduplication latency to as low as the application-aware local deduplication while saving as much

cloud storage cost as the application-aware global deduplication. Our application- aware deduplication design is motivated by the systematic deduplication analysis on personal storage.

We observe that there is a significant difference among different types of applications in the personal computing environment in terms of data redundancy, sensitivity to different chunking methods, and independence in the deduplication process. Thus, the basic idea of ALG-Dedupe is to effectively exploit this application difference and awareness by treating different types of applications independently and adaptively during the local and global duplicate check processes to significantly improve the We propose a new metric,

‘‘bytes saved per second,’’ to measure the efficiency of different deduplication schemes on the same platform. We design an application-aware deduplication scheme that employs an intelligent data chunking method and an adaptive use of hash functions to minimize computational overhead and maximize deduplication effectiveness by exploiting application awareness.

We combine local deduplication and global deduplication to balance the effectiveness and latency of deduplication. To relieve the disk index lookup bottleneck, we provide application-aware index structure to suppress redundancy independently and in parallel by dividing a central index into many independent small indices to optimize lookup performance.. at the client side to improve data transfer efficiency by grouping many small data packets into a single larger one for cloud storage. Our prototype implementation and real dataset driven evaluations show that our ALG-Dedupe outperforms the existing state-of-the-art source deduplication schemes in terms of backup window, energy efficiency, and cost saving for its high deduplication efficiency and low system overhead.

The remainder of this paper is organized as follows: We formulate the research problem in Section 2 and conduct deduplication analysis on personal data in Section 3. We describe the detailed design of ALG-Dedupe in Section 4.We evaluate ALG-Dedupe by comparing it with the existing state-of-the-art schemes in Section 5.

2 PROBLEM FORMULATION

For a backup dataset with logical dataset size L, its physical dataset size will be reduced to PL after local source deduplication in personal computing devices and further decreased to PG by global source deduplication in the cloud, PL 9 PG. We divide the backup process into three parts: local duplicate detection, global duplicate detection and unique data cloud store. Here, the latencies for chunking and fingerprinting are included in duplicate detection latency. Meanwhile, we assume that there are average local duplicate detection latency TL, average global duplicate detection latency TG and average cloud storage I/O bandwidth B for average chunk size C, TG 9 TL. We can build models to calculate BWSL and BWSG for the average backup window size per chunk of local source deduplication based cloud backup and global source deduplication Though local deduplication can achieve several to tens times of duplicate elimination ratio R ¼ L=PL with low latency, from an empirical estimation in NEC [10], global deduplication can outperform local deduplication at 20 percent to 50 percent greater in deduplication effectiveness, and the research results in EMC [20] show that interclient data overlapping can reach up to 75 percent, though around 10 percent is more common. While the latency is always the Achilles Heel of cloud computing, and the average global duplicate detection latency per chunk TG is dozens or hundreds of times the latency of local duplicate detection TL [9]. To balance cloud storage cost saving and backup window shrinking in these two schemes, we choose local-global source deduplication, which reduce the backup window size by exploiting local resources to reduce deduplication latency and save cloud storage cost by leveraging cloud resources to improve deduplication effectiveness. It can outperform We can define a metric for deduplication efficiency to balance the cloud storage cost saving and backup window shrinking in source deduplication based cloud backup services. It is well understood that the deduplication efficiency is proportional to deduplication effectiveness that is always defined by duplicate elimination ratio R ¼ L=P, and inversely proportional to the average backup window size per chunk BWS with average chunk size C.Based on this understanding and to better quantify and compare deduplication efficiency of a wide variety of deduplication techniques, we propose a new metric, called‘‘bytes saved per second,’’ which is expressed in (6), to measure the deduplication efficiency DE of different deduplication schemes on the same platform. Our local global source deduplication design can achieve high efficiency for its global deduplication effectiveness and reduced backup window. Different from the traditional deduplication based cloud backup services that are oblivious to the file level semantic knowledge, we optimize the efficiency for the source deduplication based cloud backup services by exploiting application awareness. We can divide backup dataset into n application data subsets according to file semantics, and improve the deduplication effectiveness and decrease deduplication
latency by dedicated deduplication process for each kind of application data. For application i, we define Li, PLi, and PGi as its logical data subset size, its physical data subset sizes after traditional local and global source deduplication, respectively; PALi and PAGi as its physical data subset sizes after local and global application aware source deduplication, respectively. We assume its average latency per chunk for local and global application aware duplication detection are TALi and TAGi, respectively, then TALi G TL and TAGi G TG due to the index lookup optimization by exploiting application awareness. According to our observation in Section 3: the amount of data shared among different types of applications is negligibly small, we have PLi _ PALi and PGi _ PAGi for any i is established. We make formulas to estimate the upper bound of physical dataset size PAG and the average backup window size per chunk BWSALG for application-aware local-global source deduplication based cloud backup services, and then we know formula is established, and it indicates that the efficiency of application-aware local-global source dedudeduplication DEALG is higher than that of local-global source deduplication DELG.

3 DEDUPLICATION ANALYSIS ON PERSONAL DATA

In this section, we will investigate how data redundancy,

DOI: 10.18535/ijecs/v5i6.17

space utilization efficiency of popular data chunking Methods and computational overhead of typical hash Functions change in different applications of personal Computing to motivate our research. We perform preliminary experimental study on datasets collected from desktops in our research group, volunteers' personal laptops, personal workstations for image processing and financial analysis, and a shared home server. Table 1 outlines the key dataset characteristics: the number of devices, applications and dataset size for each studied workload. To the best of our knowledge, this is the first systematic deduplication analysis on personal storage.

Observation 1

A disproportionally large percentage of storage space is Occupied by a very small number of large files with very Low chunk-level redundancy after file-level dedupe. Implication.File-level deduplication using weak hash fun-ctions for these large files is sufficient to avoid hash collisions for small datasets in the personal computing environment. To reveal the relationship between file count and storage capacity under variousfiles size.

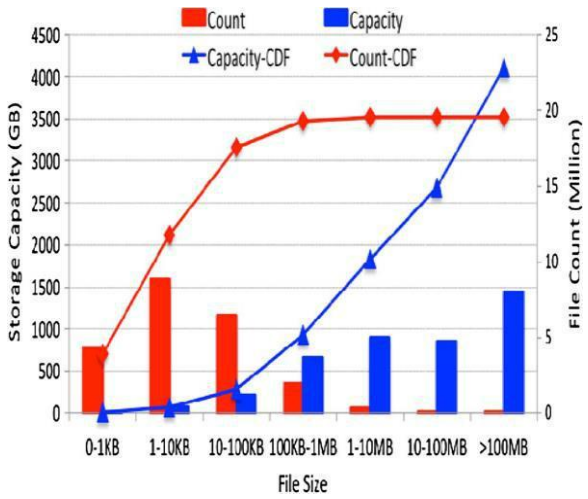

Fig. 1. Distribution of capacity and count as a function of file size. The histograms are the values of discrete density functions on file count and capacity, while the lines are cumulative distribution functions for them.

We observe that about 60.3 percent of all files are smaller than 10 KB, accounting for only 1.7 percent of the total storage capacity, and only 1.5 percent files are larger than 1MB but occupy 77.2 percent of the storage capacity. These results are consistent with. This suggests that tiny files can be ignored during the deduplication process as so to improve the deduplication efficiency, since it is the large files in the tiny minority that dominate in determining the deduplication efficiency. In the datasets mentioned above, we also find that compressed files larger than 1 MB occupy 61.2 percent storage space. To verify the data redundancy in the compressed files, we carried out chunk-level deduplication using two popular methods: Static Chunking (SC) [11] of 4 KB chunk size

and TTTD based Content Defined Chunking (CDC) [of 4 KB average chunk size (Min: 2 KB, Max: 16 KB) after file level deduplication in about 2.6TB data of typical PCapplications using compression, respectively. It shows the chunk-level data redundancy after file-level deduplication in typical application groups. Here, according to the function of applications, we group file types using compression into application groups: video, audio, image, Linux-AC for compressed archive file types in Linux, Mac-AC for compressed archive file types in Mac OS X, Windows-AC for compressed archive file types in Windows.

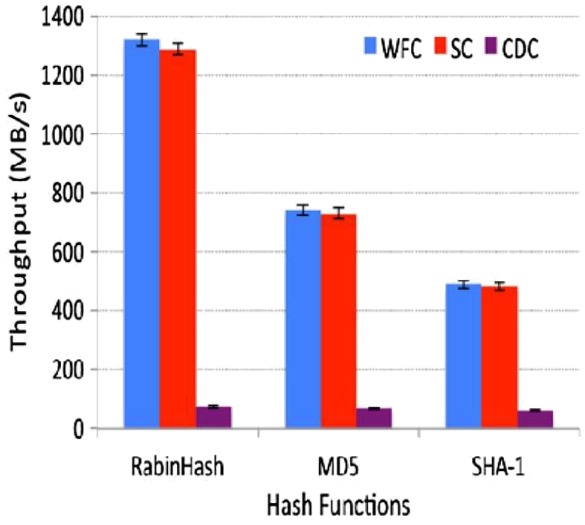

Fig. 2. Throughput of chunking and fingerprinting

Time is spent on the hash calculation itself. CDC-based deduplication has the lowest throughput on chunking and fingerprinting because most of its computational Overhead is on identifying the chunk boundaries instead of chunk fingerprinting. The deduplication strategy based on simpler chunking schemes (e.g., WFC or SC) can achieve a higher throughput because of their lower metadata storage and chunking overheads, while deduplication strategies with weaker hash functions (e.g., Rabin hash) obtain a higher throughput because of their lower computational overhead. Furthermore, the combined response time of Rabin hash and MD5 is even less than that of SHA-1. This suggests that we can employ the extended Rabin hash value as chunk fingerprint for local duplicate detection and MD5 for global duplicate detection on compressed files to reduce the computational overhead with low probability of hash collision in both small PC dataset and large-scale cloud dataset. We use SC-based deduplication with the SHA-1 or CDC-based deduplication with MD5 for both local and global deduplication on those uncompressed application datasets. Observation 3.The amount of data shared among different types of applications is negligibly small due to the difference in data content and format in these applications.

Implication

Application-aware deduplication has a potential to improve the efficiency of deduplication by eliminating Redundancy in each application independently and in Parallel. We first made this proposition in our primary study with empirical observations and analysis [21], which was subsequently

confirmed by a recent
Microsoft Research's paper [18] in their datasets from
15 globally, distributed servers. To guide our application-aware deduplication design, we conduct a content overlapping analysis to exploit the independent
parallel local-global deduplication among the clients and in the cloud. We first examine the data redundancy of intra-application and inter-application by measuring the space savings of deduplication within applications and across applications. To discover the data redundancy, we chunk files with a fixed chunk size of 4 KB and calculate the corresponding MD5 value as the chunk fingerprint in each application dataset. We first compare fingerprints in each application for intra-application Redundancy, then compare fingerprints between any two applications to identify the overlapping data between these applications for inter-application redundancy in all datasets. As seen in Table 3, we find that the loss in deduplication savings is negligibly small for all datasets when partitioning application dataset by file type and only performing intra-application deduplication. So the amount of shared data among the application groups is negligibly small due to the difference in data content and format in application datasets, which makes independent parallel deduplication among different application groups possible. As a result, the full fingerprint index can be divided into small independent indices according to the data type information in different applications, enabling it to greatly benefit from small indices to avoid on-disk index lookup bottlenecks [14], [15] by leveraging data locality in applications to prefetch appropriate application indices into memory, while exposing higher index access parallelism with low lock contention on chunk index structure.

Observation 4

To exploit chunk-level redundancy, the best choices of chunking method and chunk size to achieve high deduplication efficiency vary with different application datasets.

Implication

For each application data subset, dedicated deduplication design can significantly improve deduplication efficiency over traditional deduplication schemes with single chunking method and solely chunk size for all application types.To discover high chunk-level redundancy, we need to choose chunking method and chunk size to strike a good balance between the capability of redundancy discovery and the deduplication overhead. We always use SC-based
or CDC-based deduplication schemes. The effectiveness of the former lies in its simplicity in splitting files into small chunks with a fixed chunk size. The latter partitions data into variable size chunks based on the data content rather than the data position to avoid the chunk boundary shifting problem [13] caused by data updates with high computational overhead. It is also important to select chunk size since poor chunk size selection harms efficiency: too large chunk size reduces the exploitable redundancy in datasets, while too small chunk size can greatly increase the overhead of representing and transferring the datasets. We test the efficiency of local-global source deduplication based cloud backup service on 3 applications

Datasets with high redundancy: 160 GB Linux kernel Source code (Linux), 313 GB Virtual Machine disk images (VM) and 87 GB Microsoft Word documents with multiple versions (DOC), as a function of chunking method and chunk size. The results are shown in Fig. 3 with 4.3 MB/s mean upload bandwidth and about 300 ms average cloud latency for duplicate detection. We observe that the optimal chunk size for the highest deduplication efficiency varies among different application types. Hence, our application-aware deduplication design can significantly improve the deduplication efficiency for each application. data subset by adaptively selecting chunking method and chunk size according to dataset characteristics.

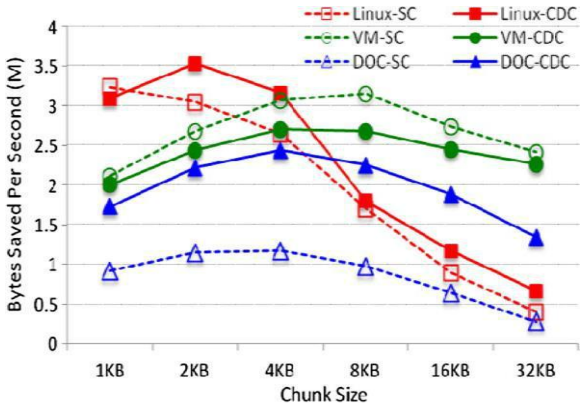

Fig. 3. Difference of deduplication efficiency as a function of chunk size and chunking method for various applications. Linux dataset can achieve highest efficiency by CDC scheme with 2 KB average chunk size, VM dataset reaches best efficiency by SC method with 8 KB chunksize, and DOC dataset can get peak efficiency by CDC scheme with 4 KB

4.1 Architecture Overview

An architectural overview of ALG-Dedupe is illustrated in Fig. 4, where tiny files are first filtered out by file size filter for efficiency reasons, and backup data streams are broken into chunks by an intelligent chunker using an application aware chunking strategy. Data chunks from the same type of files are then deduplicated in the application-aware deduplicator by generating chunk fingerprints in hash engine and performing data redundancy check in application-aware indices in both local client and remote cloud. Their fingerprints are first looked up in an application-aware local index that is stored in the local disk for local redundancy check. If a match is found, the metadata for the file containing that chunk is updated to point to the location of the existing chunk. When there is no match, the fingerprint will be sent to the cloud for further parallel global duplication check on an application- aware global index, and then if a match is found in the cloud, the corresponding file metadata is updated for duplicate chunks, or else the chunk is new. On the client side, fingerprints will be transferred in batch and new data chunks will be packed into large units called segments in the segment store module with tiny files before their transfers to reduce cloud computing latency and improve network bandwidth efficiency over WAN. On the cloud

datacenter side, segments and its corresponding chunk fingerprints are stored in self describing data structureVcontainerVin cloud storage, supported by the parallel container store. We will now describe the deduplication process in more detail in the rest of this section.

4. File Size Filter

Most of the files in the PC dataset are tiny files that less than 10 KB in file size, accounting for a negligibly small percentage of the storage capacity. As shown in our statistical evidences in Section 2, about 60.3 percent of all files are tiny files, accounting for only 1.7 percent of the total storage capacity of the dataset. To reduce the Metadata overhead, ALG-Dedupe filters out these tiny Files in the file size filter before the deduplication process, and groups data from many tiny files together into larger units of about 1 MB each in the segment store to increase the data transfer efficiency over WAN.

4.1 Intelligent Data Chunking

The deduplication efficiency of data chunking scheme among different applications differs greatly as we discussed in Section 2. Depending on whether the file type is compressed or whether SC can outperform CDC in deduplication efficiency, we divide files into three main categories: compressed files, static uncompressed files, and dynamic uncompressed files. The dynamic files are always editable, while the static files are uneditable in common.Some examples are shown in Fig. 5. To strike a better

Intelligent Data Chunking

The deduplication efficiency of data chunking scheme among different applications differs greatly as we discussed in Section 2. Depending on whether the file type is compressed or whether SC can outperform CDC in deduplication efficiency, we divide files into three main categories: compressed files, static uncompressed files, anddynamic uncompressed files. The dynamic files are always editable, while the static files are uneditable in common. Some examples are shown in Fig. 5. To

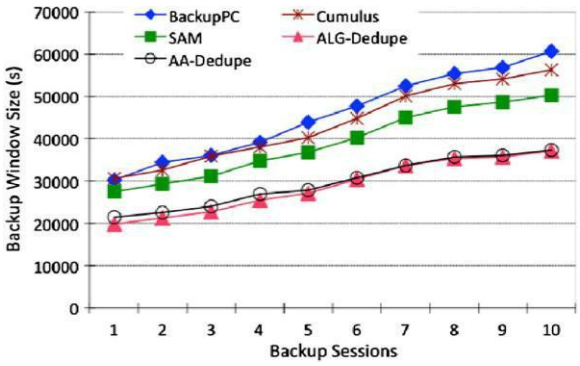

Fig 4.Application-aware index structure.

Our experimental results in Fig. 6 present the cumulative cloud storage capacity required of the providers at each backup session for individual user with the six cloud backup schemes. Different from source deduplication schemes, Jungle Disk fails to achieve high cloud storage saving due to the fact that its incremental backup scheme cannot eliminate file copies written in different places. In the source deduplication schemes, the coarse-grained method BackupPC cannot find more redundancy than other fine-grained mechanisms. The fine-grained Cumulus only performs local duplicate check, and limits the search for unmodified data to the chunks in the previous versions of the file, so it achieves lower space saving than the localdeduplication- only application-aware deduplication AADedupe

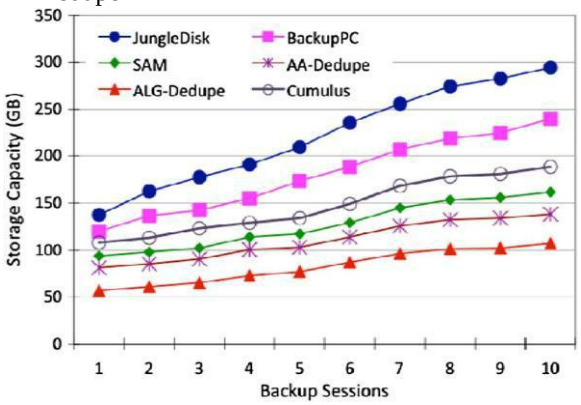

Fig. 5. Cloud storage space requirement.

The high effectiveness of data deduplication of the fine grained or global deduplication schemes comes at a significant overhead that throttles the system throughput. In our ALG-Dedupe, we perform parallel local duplication detection on shared hash-table based application-aware index structure that is stored in RAM at client side. For high parallel global duplication check in SimpleDB, we apply horizontal partitioning to divide the whole unclassified index into many small independent domains that are partitioned by file-type directed application grouping. Despite of the high WAN latency, we can significantly improve the global deduplication performance of ALGDedupe by batch I/O and parallel query. We present a comparison of the five cloud backup schemes in terms ofdeduplication efficiency in Fig. 7, and employ our proposed new metric of ‘‘bytes saved per second’’, defined in Section

2.3, to measure the efficiency of different deduplication approaches in the same cloud storage platform. ALG-Dedupe perform much better than other backup schemes in the deduplication efficiency measure with a low overhead. This significant advantage of ALGDedupe is primarily attributed to its application awareness and global duplicate detection in the deduplication process. We observe that the deduplication efficiency of ALGDedupe is 14 percent higher than our previous local scheme AA-Dedupe, owing to its advantage in global design, about 1.6 times that of the application-oblivious SAM and 1.9 times that of the local-deduplication Cumulus, more than 2.3 times that of the coarse-grained BackupPC on

DOI: 10.18535/ijecs/v5i6.17

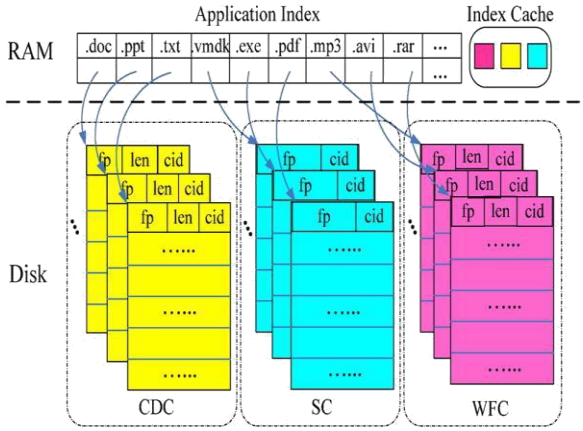

Fig. 6. Backup window size of 10 backup sessions.

5.CONCLUSION AND FUTURE WORK

In this paper, I have proposed an open evaluating plan for the recovering code-based distributed storage framework, where the information proprietors are advantaged to assign TPA for their information legitimacy checking. To ensure the first information protection against the TPA, we randomize the coefficients in the first place rather than applying the visually impaired procedure amid the evaluating procedure. Considering that the information proprietor can't generally stay online in rehearse, so as to keep the capacity accessible and unquestionable after a noxious debasement, we present a semi-trusted intermediary into the framework display and give a benefit to the intermediary to handle the reparation of the coded pieces and authenticators. To better proper for the recovering code-situation, the authenticator's BLS signature is taken into account. This authenticator can be effectively produced by the information proprietor at the same time with the encoding method. Broad investigation demonstrates that the plan is provable secure, and the execution assessment demonstrates that

our plan is exceedingly productive and can be plausibly incorporated into a recovering code-based cloud capacity framework.In this system each document is stored under specific cluster to make the maintenance and the retrieval easy. In Phase I the data upload with security and encryption and data storage under specific cluster

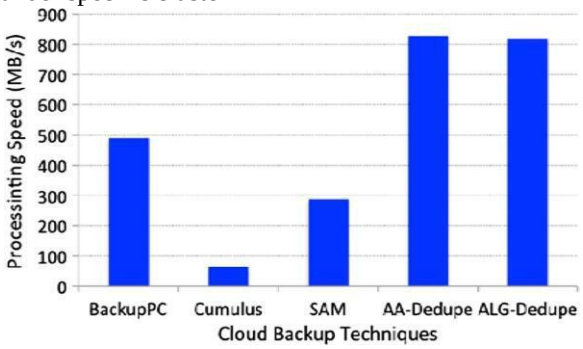

Fig. 7. Speeds of chunking and fingerprinting in PC clients.

ACKNOWLEDGMENT

This research was supported in part by the 863 Program of China under Grant 2013AA013201, the National Natural Science Foundation of China under Grant 61025009, 61232003, 61120106005, 60903040, 61070198 and 61170288, China Scholarship Council, and the US NSF under Grants CCF-0937993, IIS-0916859, CNS-1016609 and CNS-1116606. N. Xiao is the corresponding author. A preliminary version of the paper was presented at the 2011 IEEE Cluster Conference.

REFERENCES

[1] M. Armbrust, A. Fox, R. Griffith, A.D. Joseph, R. Katz,A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia,
''A View of Cloud Computing,'' Commun. ACM, vol. 53, no. 4, pp. 49-58, Apr. 2010.

[2] H. Biggar, ''Experiencing Data De-Duplication: Improving Efficiency and Reducing Capacity Requirements,'' Enterprise Strategy Grp., Milford, MA, USA, White Paper, Feb. 2007.

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

[4] A. Katiyar and J. Weissman, ''ViDeDup: An Application-Aware Framework for Video De-Duplication,'' in Proc. 3rd USENIX Workshop Hot-Storage File Syst., 2011, pp. 31-35.

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

[6] BackupPC, 2011. [Online]. Available: http://backuppc.sourceforge.net/

[7] A. Muthitacharoen, B. Chen, and D. Mazie` res, ''A Low-Bandwidth Network File System,'' in Proc. 18th ACM SOSP,2001, pp. 174-187.

[8] EMC Avamar, 2011. [Online]. Available: http://www.emc.com/avamar

[9] S. Kannan, A. Gavrilovska, and K. Schwan, ''Cloud4HomeVEnhancing Data Services with @Home Clouds,'' in Proc. 31st ICDCS, 2011, pp. 539-548.

[10] Maximizing Data Efficiency: Benefits of Global Deduplication-NEC, Irving, TX, USA, NEC White Paper, 2009.

[11] D. Meister and A. Brinkmann, ''Multi-Level Comparison of Data Deduplication in a Backup Scenario,'' in Proc. 2nd Annu. Int'l SYSTOR, 2009, pp. 1-8.

[12] D. Bhagwat, K. Eshghi, D.D. Long, and M. Lillibridge, ''Extreme Binning: Scalable, Parallel Deduplication for Chunk Based File Backup,'' HP Lab., Palo Alto, CA, USA, Tech. Rep. HPL-2009-10R2, Sept. 2009.

[13] K. Eshghi, ''A Framework for Analyzing and Improving Content Based Chunking Algorithms,'' HP Laboratories, Palo Alto, CA,USA, Tech. Rep. HPL-2005-30 (R.1), 2005.

[14] B. Zhu, K. Li, and H. Patterson, ''Avoiding the Disk Bottleneck in the Data Domain Deduplication File System,'' in Proc. 6th USENIX Conf. FAST, Feb. 2008, pp. 269-282.

[15] 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. 7th USENIX Conf. FAST, 2009, pp. 111-123.

[16] P. Anderson and L. Zhang, ''Fast and Secure Laptop Backups With Encrypted De-Duplication,'' in Proc. 24th Int'l Conf. LISA,2010, pp. 29-40.

[17] Jungle Disk. 2011. [Online]. Available: http://www.jungledisk.com/

[18] A. El-Shimi, R. Kalach, A. Kumar, J. Li, A. Oltean, and S. Sengupta,''Primary Data DeduplicationVLarge Scale Study and System Design,'' in Proc. USENIX ATC, 2012, pp. 285-296.

[19] P. Shilane, M. Huang, G. Wallace, and W. Hsu, ''WAN Optimized Replication of Backup Datasets Using Stream- Informed Delta Compression,'' in Proc. 10th USENIX Conf.FAST, 2012, pp. 49-64.

[20] F. Douglis, D. Bhardwaj, H. Qian, and P. Shilane, ''Content-Aware Load Balancing for Distributed Backup,'' in Proc. 25th USENIX Conf. LISA, Dec. 2011, pp. 151-168.

[21] Y. Fu, H. Jiang, N. Xiao, L. Tian, and F. Liu,13th IEEE Int'l Conf. CLUSTER Comput., 2011, pp. 112-120.