Similarity And Location Aware Scalable Deduplication System For Multimedia Storage Systems

G.Ajantha[1], T.K.Revathi[2]
[1, 2] Department of CSE
[1, 2] Kongunadu college of engineereing and Technology

Abstract- *Deduplication is an approach of eliminate accumulated data blocks with equal content, and has been exposed to effectively diminish the disk gap for storing large gigabyte of virtual machine (VM) images. However, it remainders challenging to classify deduplication in a real system, such as storage platform, where VM images are frequently inserted and retrieved. One of the main challenges in cloud computing is due to increasing demand of virtual machine Image storage. Existing system strives to decrease the storage consumed by virtual machine images. So in this paper proposed a SILO framework to implement the deduplication scheme in equally file and block level. And suggest a deduplication file system with short storage consumption and high-performance IO, which gratifies the requirements of VM hosting. Finally we extend our approach to implement multimedia files to check the deduplication in real time environments. And using heart beat protocol to analyze the data losses in data server and provide improved backup system.*

Keywords- Deduplication, Storage area network, Meta server, Backup system

I. INTRODUCTION

In computing, data deduplication is a data compression technique for reducing redundant copies of repeating data. It can be related to multiple instances of data. This system is used to progress storage utilization and can also be practical to network data moves to decrease the number of bytes that must be sent.

In the deduplication process, exacting chunks of data, or byte patterns, are recognized and stored during a process of analysis. As the analysis carry on, other garbage are compare to the stored copy and when an equal occurs, the needless chunk is replaced with a diminutive reference that points to the stored chunk. Given that the similar byte model may occur dozens, hundreds, or even thousands of times (the match frequency is dependent on the chunk size), the amount of data that must be accumulated or transferred can be greatly reduced.

This type of deduplication is dissimilar from that performed by normal file-compression tools, such as LZ78 and LZ77. Whereas these effort to identify small repeated substrings inside individual files, the intention of storage-based data deduplication is to examine large volumes of data and identify large sections – such as entire files or large sections of files – that are the same, in order to accumulate only one copy of it. This copy might be additionally crowded in single-file compression technique. For example a typical electronic mail system might contain 100 instances of the same 1 MB (megabyte) file attachment. Each time the email platform is reverse up; all 100 instances of the attachment are saved, requiring 100 MB storage space. With data deduplication, only one request of the attachment is actually stored; the subsequent instances are referenced reverse to the saved copy for deduplication ratio of roughly 100 to 1.

II. RELATED WORK

Xun zhao implement a framework as “Liquid” which increases the Storage utilization by eliminating the duplicate copies of the repeating or redundant data. The elimination of duplicate copies is done with the help of fingerprints. The fingerprint mechanism checks whether the data requested to be stored and the data which is already stored in the database are same. If there is a match found, then the data is considered to be redundant and if there is no match then the data can be stored without any duplication. This framework also uses Bloom Filter Array (BFA) to check whether the user has already stored the data.

J.Wei analyzed an approach better utilizes server resources, allowing many different operating system instances to run on a small number of servers, saving both hardware acquisition costs and operational costs such as energy, management, and cooling. Individual VM instances can be separately managed, allowing them to serve a wide variety of purposes and preserving the level of control that many users want.

C. Tang, seeks for a low-cost architecture option and considers that a backup service uses the existing cloud computing resource. Performing deduplication adds significant memory cost for comparison of content fingerprints. Since each physical machine in a cluster hosts many VMs, memory contention happens frequently. Cloud providers often wish that the backup service only consumes small or modest resources with a minimal impact to the existing cloud services. Another challenge is that deletion of old snapshots compete for computing resource as well, because data dependence created by duplicate relationship among snapshots adds processing complexity.

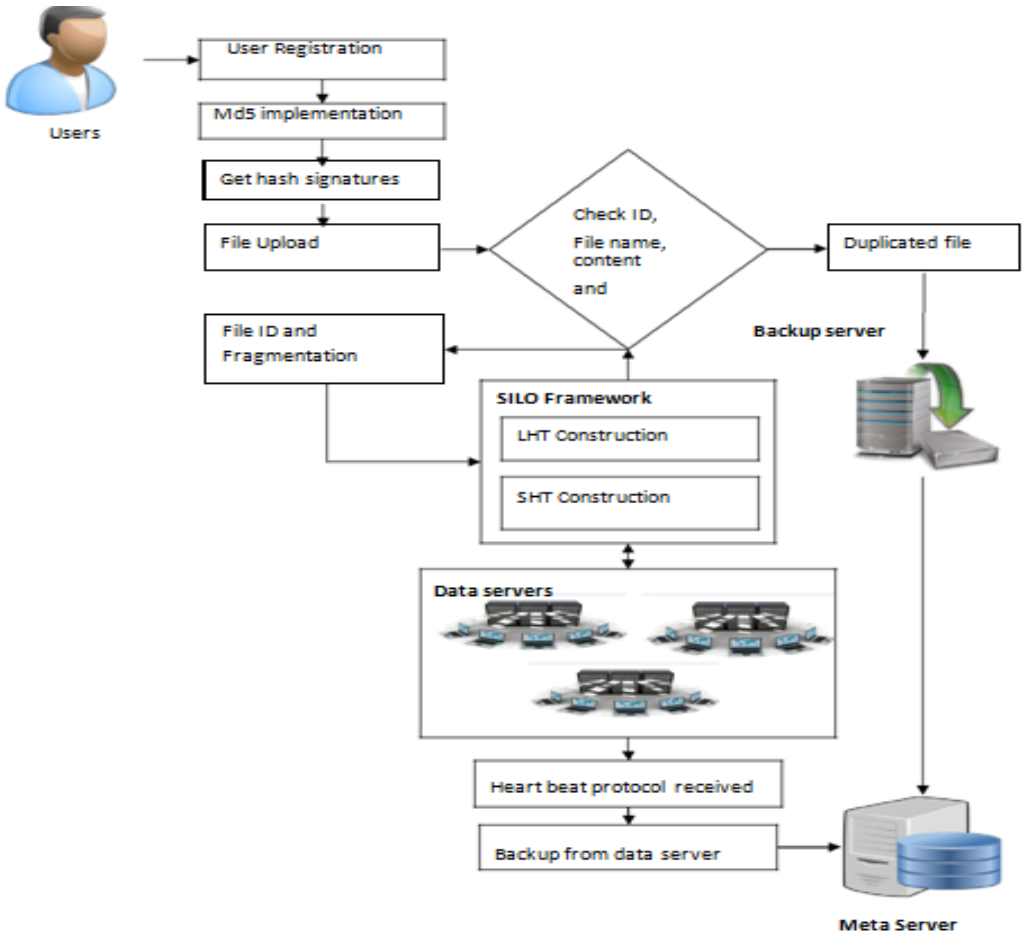
C. Ng, M. Ma makes cross-VM deduplication by excluding a small number of popular common data blocks from being backed up. Our study shows that common data blocks occupy significant amount of storage space while they only take a small amount of resources to deduplicate. Separating deduplication into multi levels effectively accomplish the major space saving goal compare the global complete deduplication scheme, at the same time it makes the backup of different VMs to be independent for better fault tolerance.

III. EXISTING METHOD

Virtual Machine is a crucial component in cloud computing environment recently. The problem in virtual machine is the demand for VM image storage. Existing systems like Storage Area Network (SAN) have taken efforts to reduce the VM image storage consumption by means of eliminating the duplicate copies. However, SAN cannot satisfy the VM deployment in large scale. File level semantics are not provided for a VM snapshot backup. Operations of snapshot takes place at the virtual device driver level and the metadata of fine-grained file system cannot be used to determine the changed data. Content fingerprints are used to develop the backup systems to identify the duplicate content. Deduplication which is done offline removes the previously written duplicate blocks during idle time. The speed of searching the duplicate fingerprints has been proposed by several techniques. Existing approaches use inline duplicate detection. Inline duplicate detection is the deduplication of an individual block which is on the write path. Waiting time can be reduced for many duplicate detection requests.

IV.PROPOSED WORK

A framework is proposed to increase the large scale VM deployment. This framework makes the VM deployment faster by using Peer-to-Peer (P2P) data transfer and reduces the storage consumption by means of eliminating the duplicate copies. Centralized architecture with multi-level and selective deduplication has been proposed. Existing set of machines hosts these services and the resource usage is in controlled manner. The snapshots within each VM undergoes deduplication process first. The searching of duplicates across VMs is a global feature that affects the parallel performance and makes the failure management complicated. The duplication is eliminated of a small data set by maintaining a deduplication ratio in a cost-effective manner. Therefore, we exploit the snapshots' data characteristics. Sharing of data across multiple VMs is reduced within this small data set and adding replicas could enhance the fault tolerance mechanism. We can extend our framework by constructing a SILO table. The proposed framework is represented in fig 1.



V. CONCLUSION

The "SILO" framework designed here can be used to provide the large scale VM deployment which may not be possible in the existing systems like Storage Area Network (SAN). This framework is a deduplication file system which provides good IO performance. This eliminates the duplicate or redundant copies of data by using fingerprint mechanism. The fingerprint mechanism can be used to identify the redundant data blocks by comparing the data block which is being requested to be stored and the data block which is already stored in the database. It then checks for a match. If a match is found then the data is considered to be redundant. This is implemented by SHT and LHT. This framework uses multimedia data transfer to increase the data availability. Data transfer reduces the time of referring to the data server every time. A node which holds the requested data can serve the data to the client and enhances the property of availability. Fault tolerance mechanism can also be implemented in this framework by using Meta server and Shadow Meta server. If the data server fails then the data can be retrieved from the Meta server and if the Meta server fails then the data can be retrieved from the Shadow Meta server. This contributes greatly to the fault tolerance mechanism.

ACKNOWLEDGEMENT

The authors wish to thank the reviewers for their valuable feedback that resulted in an improved paper and we would also like to thank our guide for assisting with code development.

REFERENCES

[1] Xun zhao et.al., ''Liquid: A Scalable Deduplication File System for Virtual MachineImages,'' in Proc. IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS., 2014, VOL 25, p.5.

[2] A. Mathur, M. Cao, S. Bhattacharya, A. Dilger, A. Tomas, and L. Vivier, ''The New ext4 Filesystem: CURRENT STATUS and Future Plans,'' in Proc. Linux Symp., 2007, vol. 2, pp. 21-33, Citeseer.

[3] M. McLoughlin, The qcow2 Image Format, Sept. 2011. [Online]. Available: http://people.gnome.org/ markmc/qcow-image-format.html

[4] C. Ng, M. Ma, T. Wong, P. Lee, and J. Lui, ''Live Deduplication Storage of Virtual Machine Images in an

Open-Source Cloud,'' in Proc. Middleware, 2011, pp. 81-100.

[5] K. Pepple, Deploying OpenStack. Sebastopol, CA, USA: O'Reilly Media, 2011.

[6] S. Quinlan and S. Dorward, ''Venti: A New Approach to Archival Storage,'' in Proc. FAST Conf. File Storage Technol., 2002, vol. 4, p. 7.

[7] R. Rivest, The md5 Message-Digest Algorithm, Apr. 1992. [Online]. Available: http://tools.ietf.org/html/rfc1321

[8] P. Schwan, ''Lustre: Building a File System for 1000-Node Clusters,'' in Proc. Linux Symp.,2003, vol. 2003, pp. 400-407.

[9] K. Shvachko, H. Kuang, S. Radia, and R. Chansler, ''The Hadoop Distributed File System,'' in Proc. IEEE 26th Symp. MSST, 2010, pp. 1-10.

[10] I. Stoica, R.Morris, D. Karger,M.F. Kaashoek, and H. Balakrishnan, ''Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications,'' in Proc. Conf. Appl., Technol., Architectures, Protocols Comput. Commun. (SIGCOMM), New York, NY, USA, Oct. 2001, vol. 31, pp. 149-160.

[11] C. Tang, ''Fvd: A High-Performance Virtual Machine Image Format for Cloud,'' in Proc. USENIX Conf. USENIX Annu. Tech. Conf., 2011, p. 18.

[12] R. Coker, Bonnie++, 2001. [Online]. Available: http://www. coker.com.au/bonnie++/

[13] D. Eastlake, 3rd, Us Secure Hash Algorithm 1 (sha1), Sept. 2001. [Online]. Available: http://tools.ietf.org/ html/rfc3174

[14] G.DeCandia, D.Hastorun,M. ampani, G.Kakulapati, A. Lakshman, A. Pilchin, S. Sivasubramanian, P.Vosshall, andW.Vogels, ''Dynamo: Amazon's Highly Available Key-Value Store,'' in Proc. 21st ACM SIGOPS SOSP, New York, NY, USA, 2007, vol. 41, pp. 205-220.

[15] C. Dubnicki, L. Gryz, L. Heldt, M. Kaczmarczyk, W. Kilian, P. Strzelczak, J. Szczepkowski, C. Ungureanu, and M. Welnicki, ''Hydrastor: A Scalable Secondary Storage,'' in Proc. 7th Conf. File Storage Technol., 2009, pp. 197-210.

[16] A. Fox, R. Griffith, A. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, and I. Stoica, ''Above the Clouds: A Berkeley View of Cloud Computing,'' Dept. Elect. Eng. Comput. Sci., Univ. California, Berkeley, CA, USA, Rep. UCB/EECS 28, 2009.

[17] S. Ghemawat, H. Gobioff, and S.T. Leung,''The Google File System,'' in Proc. 19th ACMSOSP,New York,NY, USA, Oct. 2003, vol. 37, pp. 29-43.

[18] K. Jin and E.L. Miller, ''The Effectiveness of Deduplication on VirtualMachine Disk Images,'' in Proc. SYSTOR, Israeli Exp. Syst. Conf., New York, NY, USA, 2009, pp. 1-12.

[19] M. Juric, Notes: Cuda md5 Hashing Experiments, May 2008. [Online]. Available: http://majuric.org/software/cudamd5/

[20] J. Katcher, ''Postmark: A New File System Benchmark,'' Network Applicance Inc., Sunnyvale, CA, USA, Technical Report TR3022, Oct. 1997.