International Journal of Scientific Research in Computer Science, Engineering and Information Technology
© 2018 IJSRCSEIT | Volume 3 | Issue 6 | ISSN : 2456-3307

Secure & Efficient of Cloud Storage with Data De-Duplication

G. Geetha[1], Thersphine[2]

[1]PG Scholar, Department of Computer Science and Engineering, Prist University, Puducherry, Andhra Pradesh, India

[2]Asst.Prof Department of Computer Science and Engineering, Prist University, Puducherry, Andhra Pradesh, India

ABSTRACT

Cloud computing is the most popular technique in emerging technology. It is highly useful for building the quality cloud user applications in developing projects to solve a researching problem. Cloud storages are mainly used for storing and distributing the files in the cloud environment which are handled by the cloud storage providers for the third parties by rental service. It will be useful to store number of files but the increasing the counts in storage, and then the manager cannot be guaranteed. The user may store the number of repeating files which is meant to redundancy. The file redundancy will occupy the huge number of space or storage in the cloud and lead to time complexity for searching a single file in cloud storage. That type of redundancy will be reduced by the technique called index name servers which is denoted as (INS), it will manage the file storage, and data de- duplication, to optimizing the data, to protecting the load balance and avoid the chunking match by collecting the information from the IP providers for the cloud storage. This will improve the performance of the system by the users and it will minimizing the workload of the storage and has to distribute the files.

Keywords: Cloud Storage, De- Duplication, Hash Code, Load Balancing.

I. INTRODUCTION

Cloud storage environment is used to store data that can be offered by the event. The centralized management provides the distributed & integrated storage space. Generally the storage is available in public & private cloud deployments. Due to the increased latency deserved from accessing data across the private networks, remotely accessing block storage is not a practical solution, as a result attempting to fix the latency issue using the techniques such as caching or parallelizing the I/O results in increasing risks to data integrity. Cloud services can be divided in to three stacks. Such as Infrastructure As A Service (IAAS), Platform As A Service (PAAS) & Software As A Service (SAAS). It is used to enhance the ability to achieve business goal. In addition cloud computing has been credited with increasing competitiveness through cost reduction, greater flexibility, elasticity & optimal resource utilization. With the development of the network, the number of record assimilation blockage & waste of resource can arise as the septum processes replica & lay off.

This study uses the Index Name Server (INS) to process cloud storage function, including data de-duplication, IP information, file compression & chunk matching. Because the Index Server is an indexing engine that supports analytics & search –related task for several different products & features. Therefore, our proposed INS can achieve & enhance the storage node according to the request response transmission condition.

1. Run Length Encoding (RLE)

RLE works by reducing the physical size of a repeating string of characters. This repeating string, called run, encoded in to two bytes. The first bytes represent the number of character in the run & it is called run count. The second byte is the value of the character in the run is called the run value.

Although most RLE algorithms can't achieve the high compression ratio of the more advanced compression method, RLE is both east to complement & quick to execute.

2. Distributed Hash Table (DHT)

A Distributed Hash Table (DHT) is a class of a decentralized distributed system that provides a look up service similar to a hash table (key, value). Pairs are stored in a DHT. And any participating node can efficiently retrieve the value associated with a given key. When remap the key, the data in the DHT nodes capacity be moved to another node which would discarded the bandwidth resources.

3. Bloom Filter:

Normally, the problem with hased based approach is that they have high false positive element probability The hased approach required more memory space and also the query cost incurred is very high, so some new less memory and space consuming solution was required to reduce cost. Bloom filters often use a cheap first pass to filter out segments off a dataset that do not match a query. In order to improve the accuracy, more than one hash function will be adopted to increase different mapping points.

4. Load balancing in cloud system:

Different scheduling mechanisms have different features in order to improve the efficiency & maintain the load balancing of cloud system. Most of the research aimed at consuming different scheduling algorithms for better resource enhancement.

II. INDEX NAME SERVER (INS):

An Index Name Server (INS) has comes under the Domain Name Server (DNS) which provides for all the data by numbering the files which is easy to reference the data and to avoid the de- duplication of the files. There are three functions are mainly included in INS, which are:

a) Finger prints are switched to the current file.

b) Load balancing and maintaining the files which are stored in the database.

c) Overcoming the issues of transmission and fulfill it.

An Index Name Server has a own database for managing the files and data by using the fingerprint storage of optimizing the network file transmission. Not only for storage and also INS useful for monitoring the file system in a cloud storage which allocate the huge space for storage system.

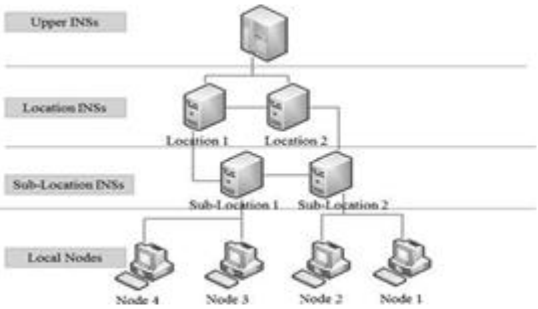

Figure 1. Hierarchical INS architecture.

In [8], the opportunistic load balancing (OLB) algorithm keeps each node busy. Thus, OLB does not consider the current workload of each node, but distributes the unprocessed tasks randomly to available nodes. Although OLB is easy and direct, this scheduling algorithm does not consider the expected task execution time and therefore cannot achieve good execution time in make span. Although the minimum completion time (MCT) algorithm gathers statistics to determine the node with the MCT, some tasks still cannot be scheduled to attain the minimum execution time (MET). The MET algorithm, on the other hand, allots the unprocessed tasks to the node with the MET, but this may cause severe load imbalance and does not suit heterogeneous network systems. Based on the MCT, the min–min scheduling [10] algorithm considers both the MCT and the MET, and assigns the tasks to the node with the MCT.

A. INS Architecture

Based on the database, the INSs adopt the stack structure of DNS, manage the storage nodes in their domain, and process users' file-access requirements. Although the INSs are similar to DNS in structure and functions, the INSs mainly query and control the data between fingerprints and storage nodes, and coordinate the transmissions by the feedback control between storage nodes and clients [1], [10], [16]. The hierarchical INS architecture is shown in Figure 1.

As displayed in Figure 2 , the INSs can be regarded as the central managers of the nodes and have server–client relation-ships [11] with one another in a hierarchical architecture to record the fingerprints and the storage nodes of all data chunks.

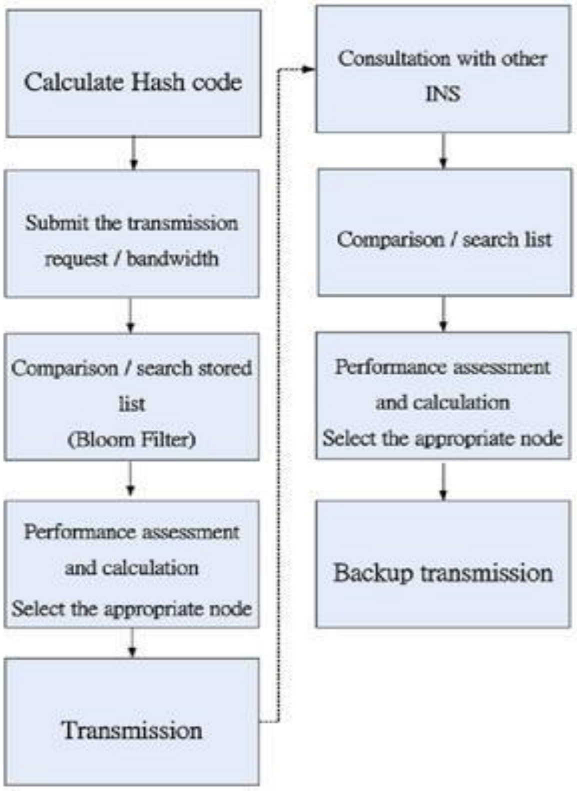
Calculate Hash code
Submit the transmission request / bandwidth
Comparison / search stored list (Bloom Filter)
Performance assessment and calculation Select the appropriate node
Transmission
Consultation with other INS
Comparison / search list
Performance assessment and calculation Select the appropriate node
Backup transmission

Figure 2. INS transmission flowchart.

B. INS Querying Process

Every domain-based INS has databases of fingerprints and storage nodes. The database of fingerprints records the finger-prints of different files and their corresponding storage nodes. When a user looks for specific fingerprints, the INS queries and confirms if the file already exists in the storage node within the domain before taking the next step. While the clients want to access data, they can use the fingerprints obtained as the index and query the INS of the upper layer, which searches for the best access node based on the content in the database in case the inefficiency of the access node or data loss. The INS transmission flowchart is shown in Figure 3.

Different requirements will lead to different query results. If the file that the client wants to access does not exist in the

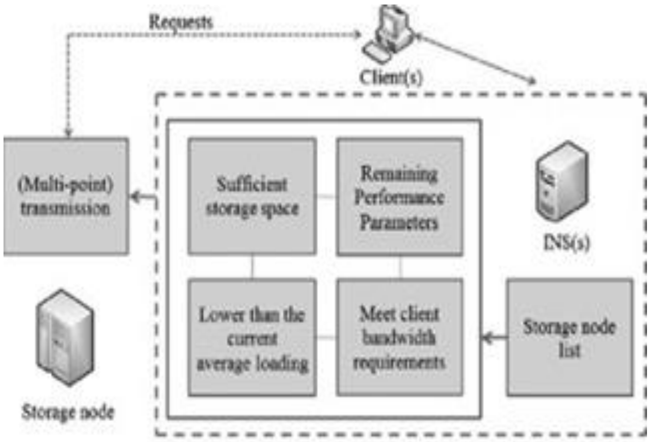
Requests
Client(s)
(Multi-point) transmission
Sufficient storage space
Remaining Performance Parameters
INS(s)
Lower than the current average loading
Meet client bandwidth requirements
Storage node list
Storage node

Figure 4. INS flowchart.

storage nodes in the local domain, the INS queries the INS of the upper layer. With the help of the Bloom Filter, the INS can find out the domain of the INS with that file chunk and also the accurate storage node through the destination INS for transmission.

III. CLOUD STORAGE FILE CHUNKING AND COMPRESSION

Structurally, the INS architecture consists of INSs, IPs, and clients, in which the INSs are responsible for controlling the whole network and handling the upload, download, and storage of data.

1) Synchronization:

The nodes that store IPs keep reporting related information to the INSs. This includes the stor-age space, the memory space, the network bandwidth, the current array number, and the surplus hardware resources. Through the information, the INSs can find the best storage nodes for clients to store data.

2) Match and lookup: Before uploading

files, the clients first send the INSs a table, which records the fingerprints of the file chunks. According to the fingerprints, the INSs can match and lookup the fingerprints already stored in the INSs.

3) Assignment:

Without determining the same fingerprints, the INSs will arrange specific IP addresses for the clients to upload files. Matching the fingerprints can accelerate data matching and delete duplicate data.

4) Transmission:

The clients transmit the files to the storage nodes assigned by the INSs and the storage nodes will report the resource spent on the task (such as CPU capacity, memory space, bandwidth, and storage space) back to the INSs for regulation and record.

IV. CONCLUSION

This paper proposed the INS to process not only file compression, chunk matching, data de-duplication, real-time feedback control, IP information, and busy level index mon- itoring, but also file storage, optimized node selection, and server load balancing. Three major contributions of this paper include the following.

1) By compressing and partitioning the files according to the chunk size of the cloud file system, we can reduce the data duplication rate. The processed files are encoded into the signature by MD5 fingerprint for the INSs to match, file, designate to the storage servers, and provide necessary uploading information for the clients. After downloading and modifying the files, the clients compress and partition the modified chunks only, encode these chunks by MD5 fingerprint and reupload the chunks.

2) According to the transmission states of storage nodes and clients, the INSs received the feedback of the previous transmissions and adjusted the transmission parameters to attain the optimal performance for the storage nodes.

3) Based on several INS parameters that monitor IP in-formation and the busy level index of each node, our proposed scheme can determine the location of

maxi-mum loading and trace back to the source of demands to determine the optimal backup node. Consequently, the backup efficiency can be improved and the load balancing among the nodes is considered.

V. REFERENCES

[1]. J. B.Connell, "A Huffman–Shannon– Fano code," Proc. IEEE, vol. 61, no. 7, pp. 1046–1047, Jul. 1973.

[2]. J. Zha, J. Wang, R. Han, and M. Song, "Research on load balance of service capability interaction management," in Proc. 3rd IEEE Int. Conf. Broadband Netw. Multimedia Technol., Oct. 2008, pp. 212–217.

[3]. R. Tong and X. Zhu,. "A load balancing strategy based on the com-bination of static and dynamic," in Proc. 2nd Int. Workshop Database Technol. Appl., Nov. 2010, pp. 1–4.

[4]. T.-Y. Wu, W.-T. Lee, Y.-S. Lin, Y.-S. Lin, H.-L. Chan, and J.-S. Huang, "Dynamic load balancing mechanism based on cloud storage," in Proc. Comput. Com. Appl. Conf., Jan. 2012, pp. 102–106.

[5]. Y. Zhang, C. Zhang, Y. Ji, and W. Mi, " A novel load balancing scheme for DHT- based server farm," in Proc. 3rd IEEE Int. Conf. Comput. Broadband Netw. Multimedia Technol., Oct. 2010, pp. 980–984.

[6]. I. Keslassy, C.-S. Chang, N. Mckeown, and D.-S. Lee, "Optimal load- balancing," in Proc. IEEE Comput. Infocom, Mar. 2005, pp. 1712–1722.

[7]. L. Zhou and H.-C. Chao. "Multimedia traffic security architecture for internet of things," IEEE Netw., vol. 25, no. 3, pp. 29–34, May 2011.

[8]. Y.-X. Lai, C.-F. Lai, C.-C. Hu, H.-C. Chao, and Y.-M. Huang, "A personalized mobile IPTV system with seamless video reconstruction algorithm in cloud networks," Int. J. Commun. Syst., vol. 24, no. 10, pp. 1375–1387, Oct. 2011.

[9]. T.-Y. Wu, C.-Y. Chen, L.-S. Kuo, W.-T. Lee, and H.-C. Chao, "Cloud-based image processing system with priority- based data distribution mechanism," Comp. Commun., vol. 35, no. 15, pp. 1809–1818, Sep. 2012.

[10]. M. Chen, C. M. Leung, L. Shu, and H.- C. Chao, "On multipath balancing and expanding for wireless multimedia sensor networks," Int. J. Ad hoc Ubiquitous Comput., vol. 9, no. 2. pp. 95–103, Feb. 2012.

[11]. Z. Feng, B. Bai, B. Zhao, and J. Su, "Redball: Throttling shrew attack in cloud data center networks," J. Internet Technol., vol. 13, no. 4, pp. 667–680, Jul. 2012.

[12]. D. Han and F. Feng, "Research on self- adaptive distributed storage system," in Proc. Wireless Commun. Netw. Mobile Comput., Oct. 2008, pp. 1–4.

[13]. J. Wang, P. Varman, and C. Xie, "Avoiding performance fluctuation in cloud storage," in Proc. Int. Conf. High Performance Comput., Dec. 2010, pp. 1–9.