Improving P2P in Cloud Computing based on Performance

Sarada.B.S
Department of Information Technology
Hindustan Institute of Technology and Science, Chennai, India

Nagarajan.S
Department of Information Technology
Hindustan Institute of Technology and Science, Chennai, India.

ABSTRACT
Cloud computing platform is a set of scalable large-scale data server clusters, it provides computing and storage services to customers. The basic study about the architecture of current cloud computing system shows that it's a central structured one; i.e. All the data nodes are indexed by a master server, but when the number requests increases it may become bottle neck of the system. This research paper is about a cloud storage architecture based on P2P with fault tolerance. Here the central entity is removed and all servers are interconnected to form a ring structure. When one of the servers fails, the work will be taken over by any of the best performing servers. The confidentiality and integrity of data passed in between the servers is also maintained using MAC algorithm.

Keywords

Cloud computing, fault tolerance, P2P, storage.

1. INTRODUCTION
Cloud computing is referred to as fifth generation computers. It sounds like a great working environment for non-technical people, as it takes away the burden of installing software's, increasing the memory capacity. It is a pay for use system, as it helps the users to access any servers from the pervasive network. It takes way the burden of managing huge networks. Cloud computing is a computing paradigm shift where computing is moved away from personal computers or an individual server to a "cloud" of computers. This method of distributed computing is done through pooling all computer resources together and being managed by software rather than a human. Cloud computing lets you access all your applications and documents from anywhere in the world, freeing you from the confines of the desktop and making it easier for group members in different locations to collaborate. With cloud computing, the software programs that are used aren't run from own personal computer, but are rather stored on servers accessed via the Internet.

2. LITERATURE SURVEY

Cloud computing is a term used to describe both a platform and type of application [3]. A cloud computing platform dynamically provisions, configures, reconfigures, and deprovisions servers as needed. Servers in the cloud can be physical machines or virtual machines. Advanced clouds typically include other computing resources such as storage area networks (SANs), network equipment, firewall and other security devices. Cloud computing also describes applications that are extended to be accessible through the Internet. These cloud applications use large data centers and powerful servers that host web applications and web services. Anyone with a suitable Internet connection and a standard browser can access a cloud application.

The key features of the cloud computing are:

- The cloud computing design is done in such a way that the resources present in it will be available from anywhere at anytime.
- Since replication of data is done in cloud computing, the resources are available even during hardware failure.
- Cloud computing provides greater speed in its operation.
- The on- demand application deployment increases the resource utilization to a large extend.
- Low cost servers are available for storage and services.

Cloud computing system [2] has the capability to hold heavy load situations without much hardware support. It makes use of the virtualization concept. For client based transactions its better to store the data in cloud. By storing the data in the cloud the traditional productivity nature is preserved and also the new upcoming technologies could also be integrated. The cloud computing architecture fosters the innovative nature, then by encouraging the new technologies to develop. The cloud architecture is highly beneficial for both large scale and small scale industries. The large scale business firms also started moving to cloud due to the features like ease of availability, remote access, cost reduction.

Google File System [4] is a scalable distributed file system for large distributed data-intensive applications. It provides fault tolerance while running on inexpensive commodity hardware, and it delivers high aggregate performance to a large number of clients. While sharing many of the same goals as previous distributed file systems, our design has been driven by observations of our application workloads and technological environment, both current and anticipated that reflect a marked departure from some earlier file system assumptions. This has led to reexamine traditional choices and explore radically different design points. The file system has successfully met different storage needs. It is widely deployed within Google as the storage platform for the generation and processing of data used by our service as well as research and development efforts that require large data sets.

The weighted voting system [7] in p2p explains a decentralized mechanism for providing sequentially-consistent access to data in a partially connected computing environment. This work has been developed in the context of pervasive computing environments. In pervasive computing scenarios it is common for devices to continually arrive and depart and disconnections can be the rule rather than the exception.

In conservative approach, maintaining strong consistency guarantees and trying to improve availability and performance within those bounds. To achieve these goals and minimize management overhead, a new concept has been developed, a decentralized weighting-voting algorithm, which guarantees sequential consistency. The algorithm distributes versioned metadata along with the data and allows online reconfiguration by using the same quorums to manage both the data and the metadata.

Tapestry is an overlay location [1] and routing infrastructure that provides location-independent routing of messages directly to the closest copy of an object or service using only point-to-point links and without centralized resources. The routing and directory information within this infrastructure is purely soft state and easily repaired. Tapestry is self-administering, fault tolerant, and resilient under load.

3. EXISTING SYSTEM

Now days a single server has the capability to handle the multiple requests from the user. But the server has to process the all the requests from the user parallel, so it will lead to a hike in the processing time of the servers. This may leads to loss of data and packets may be delayed and corrupted. On doing this the server cannot process the query from the user in a proper manner. So the processing time gets increased. It may leads to traffic and congestion. To overcome these problems we are going for the concept called "cloud computing". In this cloud computing we are going to implement the chunk server to avoid these problems.

Google Inc. has a proprietary cloud computing platform which was first developed for the most important application of Google search service and now has extended to other applications. Google cloud computing infrastructure has four systems which are independent of and closely linked to each other.

They are Google File System for distributed file storage, Map Reduce program model for parallel Google applications, Chubby for distributed lock mechanism and Big Table for Google large-scale distributed database.

A GFS cluster consists of a single master and multiple chunk servers and is accessed by multiple clients. Chunk servers store chunks on local disks as Linux files and read or write chunk data specified by a chunk handle and byte range. The master maintains all file system metadata. This includes the namespace, access control information, the mapping from files to chunks, and the current locations of chunks.

When a client wants to visit some data on a chunk server, it will first send a request to the Master, and the master then replies with the corresponding chunk handle and locations of the replicas. The client then sends a request to one of the replicas and fetches the data wanted.

Google File System (GFS) to meet the rapidly growing demands of Google's data processing needs. GFS shares many of the same goals as previous distributed file systems such as performance, scalability, reliability, and availability. However, its design has been driven by key observations of our application workloads and technological environment, both current and anticipated, which reflect a marked departure from some earlier file system design assumptions. GFS client code linked into each application implements the file system API and communicates with the master and chunk servers to read or write data on behalf of the application. Clients interact with the master for metadata operations, but all data-bearing communication goes directly to the chunk servers.

3.1 peer to peer storage

It is networking is a method of delivering computer network services in which the participants share a portion of their own resources, such as processing power, disk storage, network bandwidth, printing facilities. Such resources are provided directly to other participants without intermediary network hosts or servers. Peer-to-peer network participants are providers and consumers of network services simultaneously, which contrasts with other service models, such as traditional client-server computing where the clients only consume the server's resources. P2P is a great fit for Cloud storage systems offering the much needed reliability. It provides improved reliability than the client-server cloud.

The storage policies of P2P say that:

- The cost to the P2P system will be lower if one allocates large files to unreliable peers, and assigns smaller files to reliable peers.
- Unreliable peers should be allowed to distribute less, and reliable peers should be allowed to distribute more.

4. PROPOSED SYSTEM

New cloud storage architecture based on P2P and designs a prototype system. A cluster consists of a single database and multiple chunk servers and is accessed by multiple clients. Chunk servers store chunks on local disks and read or write chunk data specified by a chunk handle and byte range. The database maintains all file system metadata. When a client wants to visit some data on a chunk server, it will first send a request, and the database then directs with the corresponding chunk handle and locations of the replicas. Hence the processing loads on servers are balanced.

The architecture consists of mainly 3 modules:

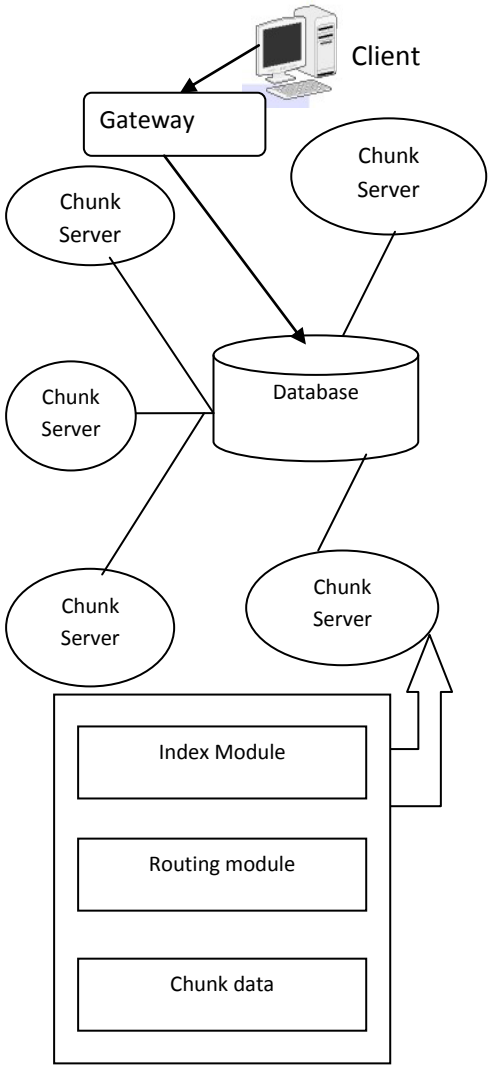

Figure 1: System Architecture

4.1 Client:

The client application is designed to get the data from the platform. Here the client sends user name and password for getting authentication. The authentication for client access is given, if and only if both the user name and password matches to one of the details in database. Else access is denied. After authentication the client send request to gateway. The client gets response after processing the request from gateway. The client can send any request to the server through the gateway. Only registered client can get the service from the server via the gateway. Thus security is implemented in client module also. Client App sends a request for a data block with logic identifier to Gateway.

4.2 Gateway:

This entity can transfer the request or response between the Client App with the network and can lead the request to the nearest node in the network. This is the important module which acts as an intermediary between the client and the Chunk server. It receives the client's request and forward the request to the nearest chunk server and then it receives the response messages from the chuck server and forward that message to corresponding client/requester. Gateway constructs a P2P search request data package including the logic ID, and sends the request to the chunk server P2P network. Gateway constructs a P2P search request data package including the logic ID, and sends the request to the chunk server P2P network.

4.3 Chunk server:

This entity is served as the data resource node and P2P node. Different with the function of pure data storage in GFS, the chunk server here has three function modules with separated interfaces. As shown in the figure above: Index Module, take charge of part of the global resource index which is assigned by DHT arithmetic such as Chord, Pastry and so on. Route Module; pass a lookup request by a next hop routing table which is also assigned by DHT. Data Module, provide the data resource stored in the local machine before a Client App can do its work, data blocks and the corresponding replica should be uploaded to the Chunk Servers. How to select the chunk servers for storage is the same with the traditional cloud computing platform.

5. CONCLUSION

We propose a cloud computing architecture based on P2P which provide a pure distributed data storage environment without any central entity for controlling the whole processing. The advantage of this is architecture is that it prevents the bottleneck problem that arises in most of the client server communications

The proposed system does its operation based on the performance of the system. It does the monitoring operation to find out the best chunk servers within the P2P network.It does this operation in order to perform efficient resource utilization and load balancing of the servers.

The future work of this proposed system could to modify the system performance by reducing the number servers present in the network. It a tough job to manage a lot number of servers. The enhancement says that if the operation is performed, for example, with the help of 100 servers, then reduce the number of servers to 50 servers by increasing the capacity of each server. Then pipelining concept could also be introduced within this P2P network in order to provide faster access. By enabling all these concepts the architecture provides better scalability, manageability, fault tolerance, better performance.

6. REFERENCES

[1] Ben Y. Zhao, John Kubiatowicz, and Anthony Joseph,"Tapestry: An Infrastructure for Fault-tolerant Wide-areaLocation and Routing", Technical Report No.UCB/CSD -01-1141, University of California Berkeley.

[2] Francesco Maria Aymerich, Gianni Fenu, Simone Surcis. An Approach to a Cloud Computing Network. 978-424426249/08/$25.00 ©2008 IEEE conference.

[3] Boss G, Malladi P, Quan D, Legregni L, Hall H. Cloud computing. IBM White Paper, 2007.

http://download.boulder.ibm.com/ibmdl/pub/software/dw/wes/hipods/Cloud_computing_wp_final_8Oct.pdf.

[4] Ghemawat S, Gobioff H, Leung ST. The Google file system. In: Proc. of the 19th ACM Symp. On Operating Systems Principles. New York: ACM Press, 2003. 29_43.

[5] Ke Xu, Meina Song, Xiaoqi Zhang,Junde Song, "A Cloud Computing Platform Based on P2P".

[6] Burrows M. The chubby lock service for loosely-coupled distributed systems. In: Proc. of the 7th USENIX Symp. on Operating Systems Design and Implementation. Berkeley: USENIX Association, 2006. 335_350.

[7] Oasis:M.Rodrig, and A. Lamarca, "Decentralized Weighted Voting for P2P Data Management," in Proc.of the 3rd ACM International Workshop on Data Engineering for Wireless and Mobile Access, pp.85–92, 2003.

[8] Antony Rowstron and Peter Druschel, "Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer systems", IFIP/ACM International Conference on Distributed SystemsPlatforms.

[9] Sigma:S. Lin, Q. Lian, M. Chen, and Z.Zhang,"A practical distributed mutual exclusion protocol in dynamic peer-to- peer systems," in Proc. of 3rd International Workshop on Peer-to-Peer Systems (IPTPS'04), 2004.