Analytical Model for Consistency on Private Cloud Storage System

Yin Nyein Aye, Thinn Thu Naing
University of Computer Studies, Yangon
yinnyeinaye.ptn@gmail.com, thinnthu@gmail.com

Abstract

Cloud computing paradigm contains many shared resources such as infrastructures, data storage, various platforms and software. In cloud storage service, consistency not only influences the performance and availability of the systems but also the overall operational cost. This paper is proposed an analytical model using MM1 queuing for data consistency on private cloud storage system which is proposed to evaluate the discarding probability based on update requests to handle update conflict .Moreover, an approach is also proposed to improve the readability after update and consistency of storage on the cloud environment.

1. Introduction

Cloud environment is a dynamic environment, where the user's data transmits from the data centre to the user's client. For the system, the user's data is changing all the time. Read and write data relating to the identity of the user authentication and permission issues.

Cloud computing is still a rather new field, which is not yet entirely defined. As a result, many interesting research problems exist, often combining different research areas such as databases, distributed systems or operating systems. There are many open research issues for cloud computing. Among these, consistency is one of the issues for cloud computing.

Cloud computing platforms enable users to rent computing and storage resources on-demand to run their networked applications and employ virtualization to multiplex virtual server's belonging to different customers on a shared set of servers. Cloud storage services are becoming increasing popular as they promise high scalability and availability at low cost. Cloud storage is an emerging infrastructure that offers Platforms as a Service (PaaS).Unfortunately, current cloud storage services are not adequate to support applications that require guarantees on consistency especially in the presence of data updates.

Cloud storage services like Amazon S3 provide weak eventual consistency guarantees, which can lead to conflicts and inconsistencies when multiple users are sharing the same cloud storage instance.[1] Current cloud storage systems have very loose consistency guarantees. Consistency plays an important role in the context of cloud computing. It is not only has a direct effect on the availability but also impacts the performance and cost. One of the most important differences between cloud storage services compared to traditional transactional database systems are the provided consistency guarantees. Data storage in cloud rely on data replication to achieve better performance and reliability by storing copies of data sets on different nodes. [6] File replication needs consistency maintenance to keep the consistency between a file and its replicas and on the other hand the overhead of consistency maintenance is determined by the number of replicas. Connecting these two components will increase the system performance.

Consistency is ensured by synchronization between the copies (replicas). This paper will address the problem of consistency on cloud storage by using probability for data consistency on cloud storage system which intends to prove the maximize resource utilization on the cloud environment. The remaining paper is organized as follows. Section 2 discusses related work and section 3 describes architecture of private cloud storage system. Moreover, section 4 represents

analytical model for consistency on cloud storage. Finally, section 5 addresses the conclusion.

2. Related Work

Cloud computing is attracting interest through the potential for low cost, unlimited scalability and elasticity of cost with load. A wide variety of offering are typically categorized as Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS). [2]The traditional database systems can become a bottleneck in a cloud platform thus novel storage platforms are commonly offered within IaaS clouds too. These storage platforms operate within the cloud platform and take advantage of the scale out from huge numbers of cheap machines; they also internally have mechanisms to tolerate the faults that are inevitable with so many unreliable machines. Although the advantages for building applications in the cloud are compelling, they come with certain limitations. Furthermore, the systems sacrifice functionality and consistency to allow for better scaling and availability. If more functionality and/or consistency are required, it has to be built on top.

A great deal of academic and industry attention has focused on the design and construction of scalable, distributed databases. Existing solutions differ in the degree of consistency they provide. High consistency implies high cost per transaction and in some situations reduced availability. Low consistency is cheaper but it might result in high operational cost because of, e.g., overselling of products in a Web shop. Moreover, .[8] is reported that not all data needs to be treat at the same level of consistency and divide into three categories (A,B,C) and treat each category differently depending on the consistency level provided.

The more replicas, the more difficult to keep consistency. When enforcing consistency, the system is no longer highly available. If the multiple copies of the same data; they must have same consistency state. In cloud computing replication is heavily utilized to maintain consistency must communicate over network but difficult when unreliable network. To maintaining consistency in clouds, updates to data require notifying all replicas of update and send messages to all replicas. If network unreliable and not all replicas respond to update, all replicas must wait. Therefore, results in unsuccessful transactions and performance degradation. [10] Amazon's S3 [1] and Dynamo [4] only promise eventual consistency. Google Storage [5] provides read-your-writes consistency. Cassandra [3] provides quorum-based consistency and eventual consistency. Voldemort [9] uses vector clocks to version data items and resolve conflicts at read time (which may require user intervention).

3. Architecture of Private Cloud Storage System

To achieve data freshness and update consistency in distributed systems, there are many possible ways of propagating updates from the data origins to intermediate nodes that have materialized views of this data. Brewer's CAP principle [7] states that any shared data system can provide only two of the following three properties: consistency, availability, and partition tolerance. Cloud-based services rely on the availability and reliability of managed data centers. The basic idea of this private cloud storage is to build a low-cost distributed cloud storage system and to provide basic data storage, read, delete, search and other storage services.

In cloud storage, it have to provide 24/7 data availability. Therefore, updating all copies synchronously is not suitable due to longer response time, especially when there are storage node failures. The storage system consists of two main stratums: a cluster based storage system and application layer. The key challenge is how to design techniques for each layer component and make these components work together in a coherent system. In figure 1 describes the architecture of each component.

At the bottom layer, replication management and storage system layers are merged in the systems relying on a fault-tolerant distributed

storage system to ensure the availability of the stored data. [6]

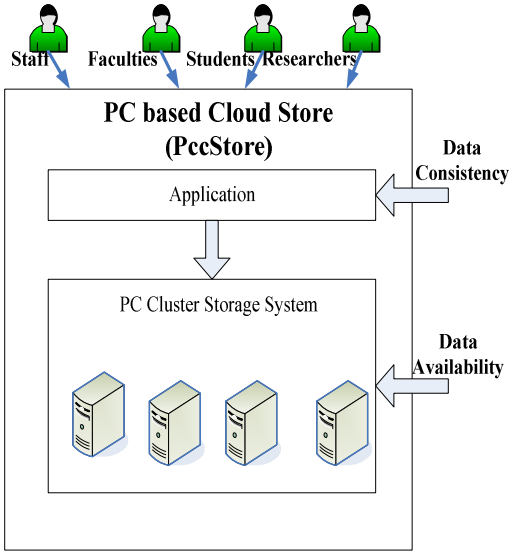

Figure 1.Architecture of Private cloud storage

The top layer of this architecture is applications of data which implements data consistency for cloud storage. From the users' point of view, data on replicated objects should appear the same as one with non-replicated system; data appears to be performed one at a time in some order. The effect of applications of data performed by clients on replicated objects should be the same as if they had been performed one at a time on a single set of objects.

In this storage system, a file is cut into small pieces of paper. The rapid recovery of data blocks and r/w rights control: A node is a data storage node, there is the possibility of failure and cannot guarantee the availability of data. However, block replica may be inaccessible due to data node unavailable. If one of the blocks is unavailable, so as the whole files' failure is normal instead of exception in cloud storage system. Still a problem is the occurrence of concurrent read and updates operations within a block. In that case, a read operation may gather blocks units that represent a different state, some may already updated and some others may still contain old content.

The same problem arises when several clients update blocks concurrently. Then data striping units may represent content of different update operations. Gathering the data stripping units over the network would then lead to invalid content. The consistency problem is similar to updating all replicas during concurrent read access.

If one replica is modified, all the replicas of an object must be updated, consuming large amounts of storage and network bandwidth. Although adding replicas of an object can improve availability and balance workload, it will increase consistency maintenance cost which may essentially result in communication congestion of underlying network. Although more replicas mean high availability and high performance, it also consumes more network resource to maintain data consistency which may lead to tremendous network resource consumption and essentially decrease the data access performance.

4. Analytical Model

Data rely on cloud storage to achieve better performance and reliability by storing copies of data sets on different nodes. In figure2 shows that consistency on distributed storage system. When a data set can be modified by applications, the problem of maintaining consistency among existing copies arises.

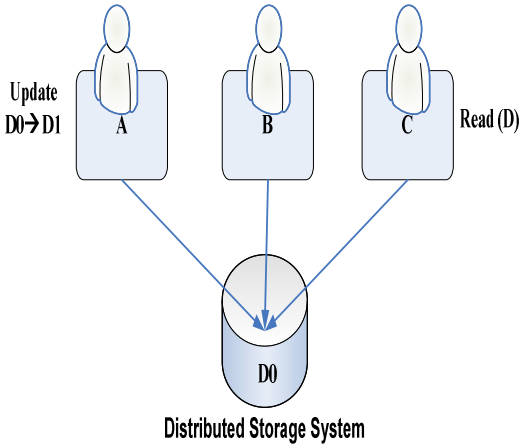

Figure 2.Consistency

Clients on different machines can access the data store concurrently; it is possible for clients to invoke conflicting operations. Two (or more) operations on the same data items are conflicting if they may occur concurrently (that is each is invoked by a different client), and at least one of them is a write. To maintain consistency, this paper is proposed MM1 queuing model for discarding probability.

Table 1.Definition of notation

Symbol	Notation
λ	Poisson arrival rate
μ	Average service rate
ρ	update request intensity
π_n	n updates in the queue
N	Number of replica nodes

An update may only be discarded by the discarding probability to control update conflict. Assuming the total number of replica nodes as N. The discarding probability of an update is computed based on the queuing model with arrival rate λ and service time μ. Define the update request intensity as ρ in equation (1)

$$\rho = \frac{\lambda}{\mu}$$

Define the probability of n updates in the queue as π_n as in equation (2) based on the queuing theory for M/M/1 finite queue.

$$\pi_n = \rho^n \pi_0$$

The discarding probability is π_n, which indicates the blocking for update request. From $\sum_{N=0}^{N} \pi_n = 1$, $\pi_0 = \frac{1-\rho}{1-\rho^N}$. The discarding probability for an update in replicas is computed in equation (3)

$$\pi_n = \frac{1-\rho}{1-\rho^N} * \rho^N$$

In this system, assume that evaluate upon different update request for arrival rate λ is 5, 10,15,20,25, service rate μ is 32 second per replica and the total number of replica nodes N is 3. When the number of update requests increase, the discarding probability also increases significantly for maintaining consistency.

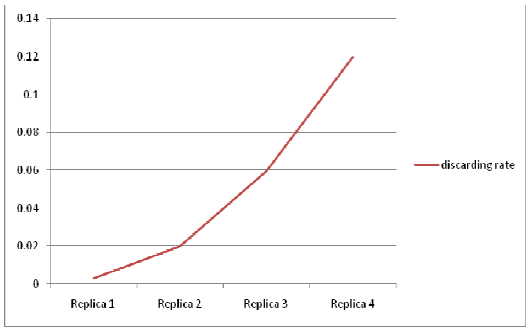

Figure 3.Discarding probability for update request

In a cluster of nodes, failures of a node (most commonly storage faults) are daily occurrences. A replica stored on a node may become corrupted because of faults in memory, disk, or network. Computers do not need to be online all the time and a file may be altered on multiple machines before they synchronize again which may result in update conflicts. After updating data, some replicas will be represented in the system with multiple versions, some of them will be active, some of them inactive because of node failure; some replicas will be up-to-date (i.e. correct), some odd (incorrect).

Consider a reading probability on cloud storage that starts in a robust state and as time progresses, it can transit to one of the four possible states S (0,0), S (1,0), S (1,1) and S (0,1) corresponding to (Inactive, wrong), (Active, wrong), (Active, correct) and (Inactive, correct). Suppose that a user will read or write request to block replica.

A correct data version will be read by cloud storage only if at least one correct replica is available. After updating replicas are active but some other inactive replicas in the system. In order to compute the probability of consistency, the state diagram of Markov chain shows the consistency on data after update.

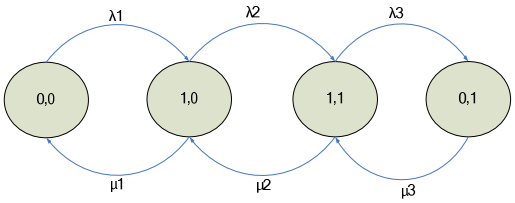

Figure 4.State diagram

Table 2 .State description for reading correct replica

State	Description
(0,0)	Inactive, Wrong
(1,0)	Active, Wrong
(1,1)	Active, Correct
(0,1)	Inactive, Correct

Write down and solving the steady-state balance equations, the probabilities as follows.

$$P(0,0) = \left[\frac{1}{1 + \frac{\lambda_1}{\mu_1} + \left[\frac{\lambda_2 + \mu_1}{\mu_2}\right] P(1,0) - \frac{\lambda_1}{\mu_2} + \left[\frac{\lambda_2 + \mu_2}{\mu_2}\right] P(1,1) - \frac{\lambda_2}{\mu_2} P(1,0)} \right]$$

$$P(1,0) = \left\{\frac{\lambda_1}{\mu_1}\right\} P(0,0)$$

$$P(1,1) = \left\{\left[\frac{\lambda_2 + \mu_1}{\mu_1}\right]\frac{\lambda_1}{\mu_1} - \frac{\lambda_1}{\mu_2}\right\} P(0,0)$$

$$P(0,1) = \left\{\left[\frac{\lambda_2 + \mu_2}{\mu_2}\right]\left[\frac{\lambda_2 + \mu_1}{\mu_2}\right]\frac{\lambda_1}{\mu_1} - \frac{\lambda_1}{\mu_1} - \frac{\lambda_2}{\mu_2}\frac{\lambda_1}{\mu_1}\right\} P(0,0)$$

Reading of a correct data is represented by a random variable C i.e. correct (C=1) and incorrect read (C=0).The correct version is going to be read, if at least one correct replica is active. In other words, it is the counter probability that Active-correct replica P (1,1) is available. The system is not available in P (0,0) and P (0,1) and P (1,0).Therefore, the probability of reading the correct read P(C=1) can be expressed for N replicas as in equation (4)

$$P(C = 1) \geq 1 - P(1,1)^N$$

If λ and μ (rate in and rate out) are equally likely for correct read replica, the read probability changes with the increase of the number of replicas.

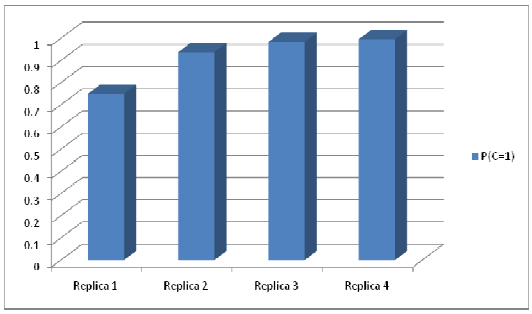

Figure 5.Correct read probability with equal rate

If λ and μ (rate in and rate out) change at any time, λ_1=0.1, λ_2=0.2, λ_3=0.3 and μ_1=0.3, μ_2=0.2, μ_3=0.1.

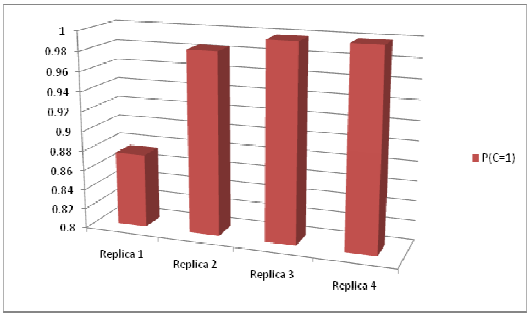

Figure 6.Correct read probability with unequal rate

In figure 5 and 6 show that the read probability changes with the increase of the number of replicas and can be seen that the system will contain at least one correct block replica.

5. Conclusion

The increasing popularity of cloud storage services has lead companies that handle critical data to think about using the services for their

storage needs. Therefore, in cloud storage services, consistency plays an important role in the context of cloud computing. It is not only has a direct effect on the availability but also impacts the performance and cost. This paper is proposed discarding probability for update request which eliminates the complicated conflict on cloud storage. Furthermore, an approach is also improved the consistency of storage on the cloud users environment.

References

[1]Amazon's Simple Storage Service(S3). http://aws.amazon.com/s3. Accessed May 2010.

[2]B. Furht, A.Escalante, Handbook of Cloud Computing, ISBN 978-1-4419-6523-3, Springer Science+Business Media, LLC 2010.

[3]Cassandra. http://incubator.apache.org/cassandra/.Accessed December 2009.

[4]Dynamo: Amazon's highly available key-value store. In SOSP'07, pages 205–220,October 2007.

[5]Google Storage for Developers. http://code.google.com/apis/storage. Accessed May 2010.

[6]Julia Myint and Thinn Thu Naing, Management of Data Replication for PC Cluster based Cloud Storage System,University of Computer Studies, Yangon, Myanmar, *IJCCSA*,Vol.1, No.3, November 2011

[7]S. Gilbert and N. Lynch. Brewer's conjecture and the feasibility of consistent, available, partition-tolerant web services. SIGACT News, 33(2):51–59, 2002.

[8]Tim Kraska , Martin Hentschel, Gustavo Alonso, Donald Kossmann, Consistency Rationing in the Cloud: Pay only when it matters, Systems Group, Department of Computer Science, ETH Zurich, VLDB '09, August 24-28, 2009, Lyon, France.

[9]Voldemort. Http://project-voldemort.com/. Accessed March 2010.

[10]Xin Chen, Haining Wang, Shansi Ren, Xiaodong Zhang, DNScup: Strong Cache Consistency Protocol for DNS.

[11]Yi Hu, Min Feng, Laxmi N. Bhuyan, A Balanced Consistency Maintenance Protocol for Structured P2P Systems,Department of Computer Science and Engineering,University of California at Riverside, CA, U.S.A.