Consistency as A Service: Trading Consistency for Availability

Pratik S. Mahajan[1] Shubhangi Sapkal[2] Dr. R. R. Deshmukh[3]
[1,3]Department of Computer Science & Engineering [2]Department of MCA
[1,2]Government College of Engineering, Aurangabad, India [3]Dr. BAMU, Aurangabad, India

Abstract— As more and more charities and libraries move to cloud services, this technology is no longer novel; it's becoming the dominant paradigm in IT. A cloud storage facility is a need of an hour as it facilitates elasticity–so resource allocation can get bigger or smaller depending on a users' demand, Elasticity enables scalability–so that the cloud can scale up when demand is high and down for less demand (e.g. allows for scale when an application in the cloud adds users, etc.), Resource Pooling, On Demand, Self-Service Provisioning i.e client can request desire service from cloud vendor and 24*7 service i.e high availability at cheapest price .But the key hindrance come when we need high availability;as it needed to create plenty of copies of single file and at different geography .To attain high consistency more money is needed. In this paper, we have presented unique, innovative, efficient and effective consistency as a service (CaaS) model, which consists of a various large capacity data clouds and small audit clouds .Data clouds are used to store clients actual data offered by Cloud vendors like Google, Amazon, Microsoft, etc .and Audit cloud are used to check consistency of cloud is maintained accurately or not. At the last we have applied traditional approach to audit purpose i.e Heuristic Auditing Strategy which uses 2 algorithm viz Local and Global Auditing Strategy. Loosely synchronized clock is needed to audit purpose. Heuristic Auditing Strategy is used to find violations in consistency terms i.e. document is modified or safe.

Key words: Audit Cloud, Consistency as a service (CaaS), Data Cloud, Global Auditing, Local Auditing

I. INTRODUCTION

Cloud Computing is still a buzzword or uncleared to many people.The definition mostly used today is the one expressed by the National Institute of Standards and Technology (NIST), which states: "a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction."[1,2].

Cloud computing comes into focus only when you think about what we always need: a way to increase capacity of system or add capabilities of system on the fly without investing in new infrastructure, training new personnel, or licensing new software. Cloud computing consists any subscription-based or pay-what-use service that, in real time over the Internet, extends ICT's existing capabilities.

Cloud computing is at an early stage, with a motley crew of providers large and small delivering a slew of cloud-based services, from full-blown applications to storage services to spam filtering. Today, for the most part, IT must plug into cloud-based services individually, but cloud computing aggregators and integrators are already emerging.

Every organization wants to use its resources well. One way to do this is by running applications in virtual machines (VMs) rather than directly on physical machines. This can be done by Virualization. Virtualization has real value, and most organizations have adopted this approach in their datacenters. Useful as it is, however, the next step in the evolution of virtualization has become clear.

There are many facilities provided by cloud computing like resources such as a virtual-machine, disk-image library, raw block storage, file or object storage, firewalls, load balancers, IP addresses, virtual local area networks (VLANs), and software bundles(IaaS),development environment to application developers(PaaS),Software as per used(SaaS);but most of the time cloud is used for storage purpose. But client needed the integrity of the data, whether the data consistent or not. Data consistency can be gained by trading availability as per the CAP theorem. The CAP Theorem can be stated as "in a network subject to communication failures, it is impossible for any web service to implement an atomic read/write shared memory that guarantees a response to every request."[3] means either we can provide availability or consistency

There are many differences in Cloud Computing and traditional transactional databases related to consistency requirement. As in cloud, there is data duplication is taken place i.e. data is geographically distributed to get higher degree of performance. But duplicated file needs strong consistency to keep the consistency between a file and its replicas and simultaneously 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.

Consider the scenario in Fig.1, where Raj and Simran working on single project together using cloud facility where data is replicated to 5 distributed data servers CS1, CS2, CS3, CS4,and CS5. After uploading a new version of the requirement analysis to a *CS*4,Simran calls Raj to download the latest version for integrated design. Here, after Simran calls Raj, the causal relationship [3] is established between Simran's update and Raj's read. Therefore, the cloud should provide causal consistency, which ensures that Simran's update is committed to all of the replicas before Raj's read. If the cloud provides only eventual consistency, then Raj is allowed to access an old version of the requirement analysis from CS5.If the old version is used,it may not satisfy the customer's actual requirement.

As per the Cloud Computing feature data should available always to the client which are replicated at multiple geographical locations. The main problem with consistency feature of cloud; it costs very high to attain strong consistency means updated copy present always to the client. So most of the cloud vendors provide weak consistency like eventual consistency, where client sees slightly old data. Eventual consistency provides very high reliability and high performance which is the requirement of domain name system. Updates to a name will not be visible immediately, but all clients are ensured to see them eventually. However, eventual consistency is not a catholicon for all applications. Especially for the interactive applications, stronger consistency assurance is of increasing importance. But different applications required different consistencies like mail services need monotonic read consistency and read your write consistency while social network need causal consistency.

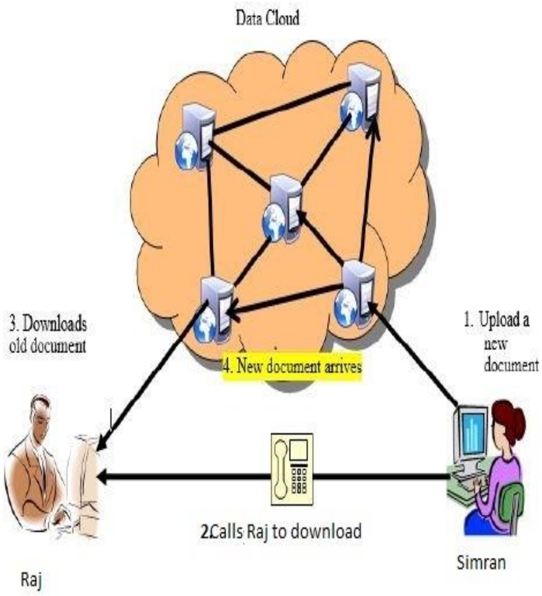


Fig. 1: An application that requires causal consistency.

II. RELATED WORK

A cloud is essentially a large-scale distributed system where each piece of data is replicated on multiple geographically-distributed servers to achieve high availability and high performance. Thus, we first review the consistency models in distributed systems. As a standard textbook, proposed two classes of consistency models: data-centric consistency and client-centric consistency. Data-centric consistency model considers the internal state of a storage system, i.e., how updates flow through the system and what guarantees the system can provide with respect to updates[5]. Data centric consistency models Concerns read and write on shared data (e.g. shared memory, shared database, distributed file system, etc.) However, to a customer, it really does not matter whether or not a storage system internally contains any stale copies. As long as no stale data is observed from the client's point of view, the customer is satisfied. Therefore, client-centric consistency model concentrates on what specific customers want, i.e., how the customers observe data updates. Their work also describes different levels of consistency in distributed systems, from strict consistency to weak consistency. High consistency implies high cost and reduced availability

Following is the Client Data Consistency Model

A. *Causal Consistency*

Definition: Causal consistency is another data centric consistency model: For a data store to be considered causally consistent, it is necessary that the store obeys the following condition:

Writes that are potentially causally related must be seen by all processes, and must be seen in the same order. Concurrent writes may be seen in a different order on different machines.

Client centric consistency models Concerns consistency experienced by any one client when accessing a distributed data store[4].

Following are the client centric consistency models :

B. *Monotonic Read*

Definition: A data store is said to provide monotonic-read consistency if the following condition holds:

If a process reads the value of a data item x then any successive read operation on x by that process: will always return that same value or a more recent value. Example: Email systemA client reading Emails by accessing a locally available replica can expect to see the same Emails when accessing another replica at a later time. It may be that new Email may arrive (and hence, may be added to the users Email database) in-between two reads. In this case, the client can expect to see all the old Emails as well as the new Emails.In other words, the Email client will never get to see an older version of the Email database when accessing Email at different replicas in the system. Such behavior is guaranteed by the monotonic read consistency model[7].

C. *Read Your Writes*

Definition: A data store is said to provide read-your-writes consistency, if the following condition holds:

The effect of a write operation by a process on data item x will always be seen by a successive read operation on x by the same process.

This is also known as the UNIX semantics.

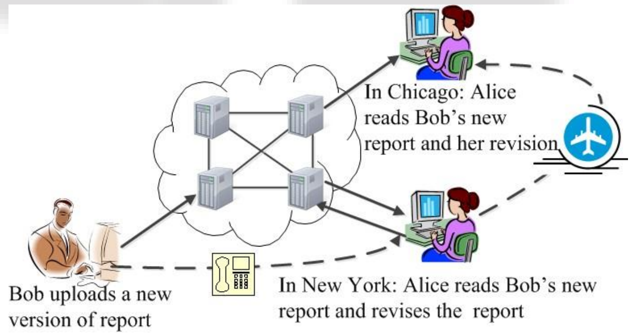


Fig. 2: An application that has different consistency requirements.

Intuitively, monotonic-read consistency requires that a user must read either a newer value or the same value, and read- your-write consistency requires that a user always reads his latest updates. To illustrate, let us consider the example in Fig.2 suppose that Alice often commutes between New York and Chicago to work, and the CSP maintains two replicas on cloud servers in New York and Chicago, respectively, to provide high availability. In Fig.2, after reading Bob's new report and revising this report in New York, Alice moves to Chicago. Monotonic-read consistency requires that, in Chicago, Alice must read Bob's new version, i.e., the last update she ever saw in New York must have been propagated to the server in Chicago. Read-your-write consistency requires that, in Chicago, Alice must read her revision for the new report,i.e., her own last update issued in New York must have been propagated to the server in Chicago. The above models can be combined. The users can choose a subset of consistency models for their applications.

III. SYSTEM DEVELOPMENT

The proposed consistency model contains Data Cloud and Audit Cloud. Data Cloud is handled by Cloud Vendor which stores file along with secret key, private key and MAC address.To provide High availability multiple replicas at multiple geographical locations is stored.An Audit cloud consists of local and global auditing strategies which is done by group of users.Auditor cloud monitors the violations of the agreement or not[6].

2 level auditing is done in the model;each users activity is recorded into User Operation Table(UOT).Each user maintains a UOT for recording local operations. Each record in the UOT is described by three elements: operation, logical vector, and physical vector. While issuing an operation, a user will record this operation, as well as his current logical vector and physical vector, in his UOT.

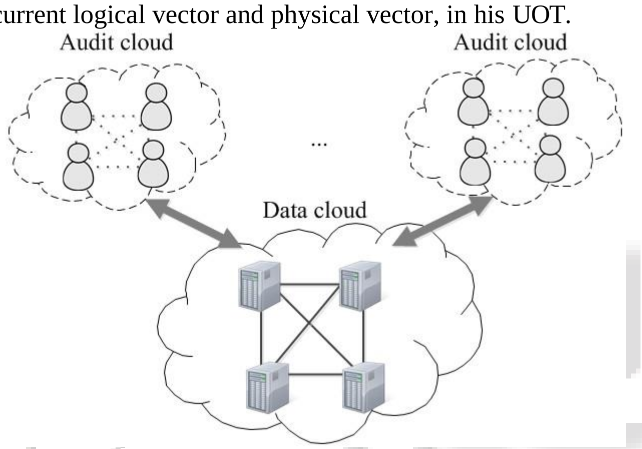


Fig. 3: System Architecture

Local auditing can be performed independently by each user with his own UOT.

A. Local Auditing Strategy

Initialize UOT with
While issue an operation op do
If op=W(a) then
Record W(a) in UOT
If op=r(a) then
W(b)ϵUOT is the last write
If W(a)->W(b) then
Read your write consistency is violated
R(c)ϵUOT is last read
If W(a)->W(c) then
Monotonic read consistency is violated
record r(a) in UOT

B. Global Auditing Strategy

All other users will send their UOTs to the auditor for obtaining a global trace of operations. After executing global auditing, the auditor will send auditing results as well as its vectors to all other users.
Each operation in the global trace is denoted by a vertex
For any 2 operations op1 and op2 do
If op1->op2 then
A time edge is added from op1 to op2
If op1 =W(a),op2=R(a),and 2 operations come from different users then
A data edge is added from op1 to op2
If op1=W(a),op2=W(b), 2 opeartions come from different users and W(a) is on the route from W(b) to R(b) then
A causal edge is from op1 to op2
Check whether the graph is DAG by Topological sorting [7].

IV. EXPERIMENTAL RESULTS

Following are the experimental results.
Data owner sends data to one of 4 cloud server and he has option to send data to auditor cloud in Fig.4. Whenever the data owner sends data to auditor auditor cloud stared its working. Data owner can add user to access his files on cloud server for causal consistency where he specifies read or write the file authentication.

Fig.5 shows 4 different cloud servers which stores the clients data and different auditing cloud for consistency check.Auditor cloud implements HAS which has local and global auditing strategies which shows staleness of data.When file is uploaded by the client at cloud server private keys and secret keys are generated for authentication purpose i.e. file can be accessed only with correct secret key.

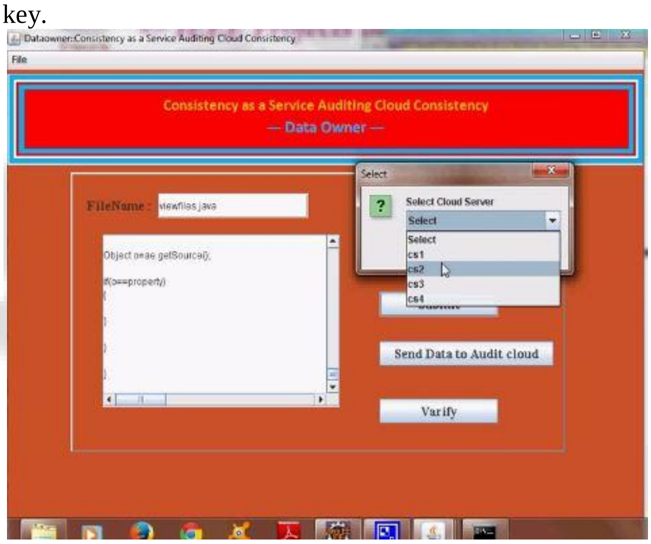
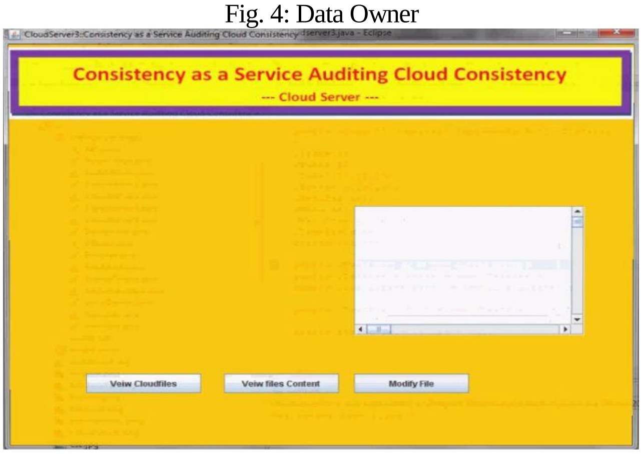


Fig. 4: Data Owner



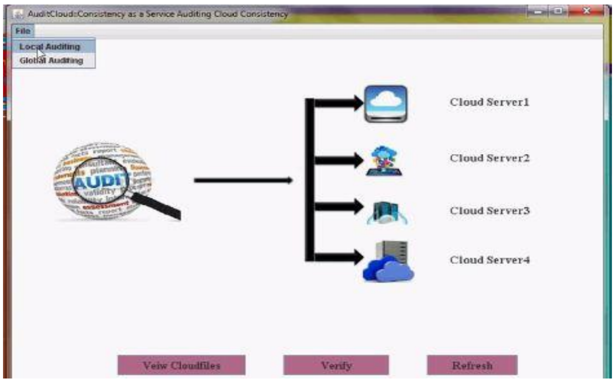


Fig. 5: Data Server and Audit Cloud

Fig.6 shows when any unidentified person try access the file auditor activated and pops the message “File is not Safe”.

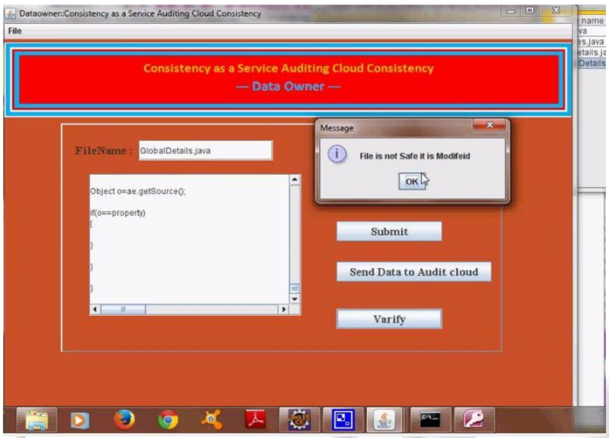


Fig. 6: File Status

At the last Auditing Report is generated whether file modified or safe,shown in Fig.7.

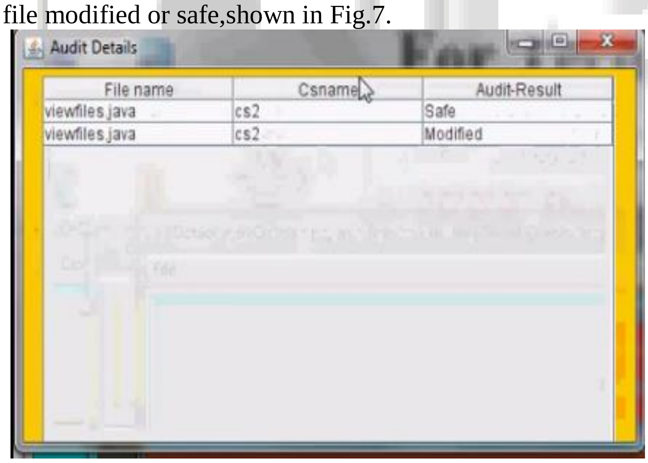


Fig. 7: Auditing Report

V. CONCLUSION

Consistency as a service (CaaS) model and a two-level auditing structure to help users validate whether the cloud service provider (CSP) is providing the promised consistency and to quantify the severity of the violations is any .With the CaaS model, the users can assess the quality of cloud services and select a right cloud service provider among various candidates, for example the least expensive one that still provides adequate consistency for the users application. . For our future work, we will conduct a thorough theoretical study of other consistency models in cloud computing and try to implement these models such as read-after-write consistency and monotonic-write consistency models practically.

ACKNOWLEDGEMENT

We thank the anonymous reviewers for their constructive suggestions and comments, which helped us to improve the content, quality, and presentation of this article.

REFERENCES

[1] M. Armbrust, A. Fox, R. Griffith, A. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, et al., “A view of cloud computing,” Commun. ACM, vol. 53, no. 4,2010.

[2] P. Mell and T. Grance, “The NIST definition of cloud computing (draft),” NIST Special Publication 800-145 (Draft), 2011.

[3] E. Brewer, “Pushing the CAP: Strategies for Consistency and Availability”, Computer, (2012) February, pp. 23-29.

[4] M. Ahamad, G. Neiger, J. Burns, P. Kohli, and P. Hutto, “Causal memory: definitions, implementation, and programming,” Distributed Computing, vol. 9, no. 1, 1995.

[5] A. Tanenbaum and M. Van Steen, Distributed Systems: Principles and Paradigms. Prentice Hall PTR, 2002.

[6] G. DeCandia, D. Hastorun, M. Jampani, G. Kakulapati, A.Lakshman, A. Pilchin, S. Sivasubramanian, P. Vosshall, and W. Vogels, “Dynamo: Amazon’s highly available key-value store,” in Proc. 2007 ACM SOSP.

[7] Qin Liu, Guojun Wang,”Consistency as A Service: Auditing Cloud”, IEEE Transactions on Network and Service Management, Vol. 11, No. 1, March 2014.