International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2013): 6.14 | Impact Factor (2013): 4.438

Checking the Data Cloud Consistency by Auditor Method

Guruprasad Nellur[1], Dr. Rekha Patil[2]

[1]Department of Computer Science and Engineering M.Tech (CSE),
Poojya Doddappa Appa College of Engineering, Gulbarga, Karnataka, India

[2]Prof.Rekha Patil, Department of Computer Science and Engineering,
Poojya Doddappa Appa College of Engineering, Gulbarga, Karnataka

Abstract*: Cloud storage services can be regarded as a typical service in the cloud computing, which involves the delivery of the data storage including database like services and network attached storage. And the cloud storage services have become very popular due to their huge advantages. To provide everywhere always-on access, a cloud service provider (CSP) maintains multiple replicas for each piece of data on geographically distributed servers. A key problem of using the replication technique in clouds is that it is very expensive to achieve strong consistency on a worldwide scale. In this paper, we present a new consistency as a service (CaaS) model, which consists of a large data clouds and multiple small audit clouds. In the CaaS model, a data cloud is maintained by a CSP, and a group of users that constitute an audit cloud can verify whether the data cloud provides the promised level of consistency or not. We also proposed two-level auditing architecture, which only requires a loosely synchronized clock in the audit cloud. Then, it designed algorithms to quantify the severity of violations with two metrics: the commonality of violations, and the staleness of the value of a read. Finally, it devise a heuristic auditing strategy (HAS) to reveal as many violations as possible. Extensive experiments were performed using a combination of simulations and a real cloud deployment to validate HAS.*

Keywords: Cloud storage, consistency as a service, two-level auditing, heuristic auditing strategy, Cloud service provider.

1. Introduction

Cloud computing is a model for enabling the ubiquitous network access to share the pool of configurable computing resources. The cloud computing focus on maximizing the effectiveness of shared resources. data storages, virtualized infrastructure, virtualized platforms, as well as software and applications are being provided and consumed as services in the cloud, e.g., per gigabyte per month. Examples include Amazon SimpleDB1, Microsoft Azure storage2, and so on. By using the cloud storage services, the customers can access data stored in a cloud anytime and anywhere, without caring about a large amount of capital investment when deploying the underlying hardware infrastructures. The main intension of this model is uploading and downloading a file from the cloud. Updates to a name will not be visible immediately, but all clients are ensured to see them eventually. Consider the following scenario as shown in Fig. 1. Suppose that Alice and Bob are cooperating on a project using a cloud storage service, where all of the related data is replicated to five cloud servers, CS1 to CS5. After uploading a new version of the requirement analysis to a *CS*4, Alice calls Bob to download the latest version for integrated design. Here, after Alice calls Bob, the causal relationship is established between Alice's update and Bob's read. Therefore, the cloud should provide causal consistency, which ensures that Alice's update is pledged to all of the replicas before Bob's read. In cloud storage, consistency not

Only determines correctness but also the actual cost per transaction. In this paper, it presents a novel *consistency as a service* (Caas) model. The CaaS model includes a large *data cloud* and multiple small *audit clouds*. The data cloud is maintained by a CSP, and an audit cloud consists of a group of users that cooperate on a job, Specifically, it require each user to maintain a logical vector for partial ordering of operations, and it adopt a two-level auditing structure: each user can perform *local auditing* independently with a local trace of operations; periodically, an auditor is elected from the audit cloud to perform global auditing with a global trace of operations. Local auditing focus on monotonic and read your write consistencies, which can be performed by a light-weight online algorithm. Finally, it proposes a heuristic auditing strategy (HAS) which adds appropriate reads to reveal as many violations as possible. The key contributions are as follows: 1) It present a new consistency as a service (CaaS) model, where a group of users that constitute an audit cloud can verify whether the data cloud provides the promised level of consistency or not. 2) It propose a two-level auditing structure, so for this it only requires a loosely synchronized clock for ordering operations in an audit cloud. 3) It designs algorithms to quantify the severity of violations with different metrics. 4) It devise a heuristic auditing strategy (HAS) to reveal as many violations as possible. Extensive experiments were performed using a combination of simulations and a real

Cloud deployment to validate HAS. The remainder of this project is organized as follows .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, our work first reviews the consistency models in distributed systems.

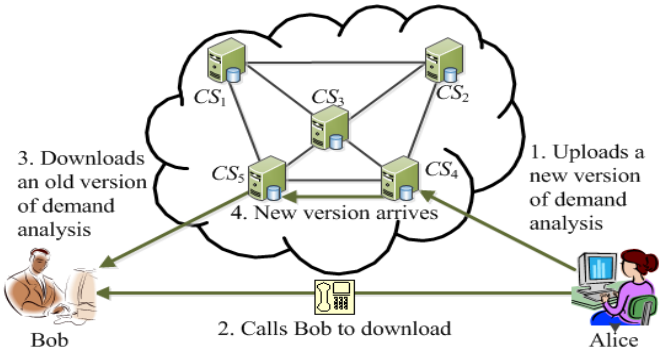


Figure 1: An application that requires causal consistency

The paper is organized, section 1 discusses the introduction, and section 2, related works. Section 3 details the system design and implementation. Section 4, presents the performance evaluations of our system design and result. Finally, section 5 presents some concluding remark.

2. Related work

"A View of Cloud Computing", M.Armbrust, R.Giffith.[1] have proposed in Cloud Computing, the long-held dream of computing as a utility, has the potential to transform a large part of the IT industry, making software even more attractive as a service and shaping the way IT hardware is designed and purchased. A cloud provider sells services at a low level of abstraction like EC2 or a higher level like App Engine; all believe that computing, storage and networking must all focus on horizontal scalability of virtualized resources rather than on single node performance. Processors should work well with VMs and flash memory should be added to the memory hierarchy, and LAN switches and WAN routers must improve in bandwidth and cost. **"Data Consistency Properties and the Trade-offs in Commercial Cloud Storages: the Consumers Perspective",Hiroshi Wada, Alan Fekete, Kevin Lee, Anna Liu.[2]** have proposed the new class of data storage systems, called No SQL (Not Only SQL), have emerged to complement traditional database systems, with rejection of general ACID transactions as one common feature. To achieve high availability and low latency, many cloud data storage platforms (or particular operations within a platform) use techniques that avoid two-phase commit and/or synchronous access to a quorum of sites. Here the risks from inconsistency seem less important compared to other sources of data corruption, such as bad data entry, operator error, customers repeating input, fraud by insiders, etc. **"Pushing the CAP: strategies for consistency and availability "Computer, vol. 45, no. 2 , 2012. [3]** Has described the, Modern large distributed Systems at wide scale have adopted new types of databases that are not subject to the features which ensure strong Consistency. In this work, it discusses the CAP theorem, its evolution and its influence on these systems. After, it talk about the misunderstandings and problems aroused by this theorem. Scalability: A system is scalable if it can grow efficiently, using new resources efficiently to handle more load. There appear to be inherent trade-offs between scalability and consistency. **"Don't settle for eventual: scalable causal consistency for wide-area storage with COPS", W.Lloyd, M.freedman, and D.Anderson [4]** have discussed about Geo-replicated, distributed data stores that support complex online applications, such as social networks, must provide an "always on" experience where operations always complete with low latency. Our evaluation demonstrates that COPS and COPS-GT provide low latency, high throughput, and scalability. **" What consistency does your key-value store actually provide", E.Anderson, X.Li, M.Shah.[5]** have discussed about the many key-value stores have recently been proposed as platforms for always-on, globally-distributed, Internet scale applications. To meet their needs, these stores often sacrifice consistency for availability. It has found that traces from Pahoehoe, an eventually consistent store, often exhibit strong consistency during failure-free runs of our cloud-based micro-benchmarks. **"Timestamps in message-passing systems that preserve the partial ordering", C.Fidge. [6]** Has discussed the Time stamping is a common method of totally ordering events in concurrent programs. However, for applications requiring access to the global state, a total ordering is inappropriate. The algorithms do not change the communications graph or require a central timestamp issuing authority. They are general in the sense that they allow any event to be compared to any other, even if they are both in the same process or both in non-neighboring processes. Nevertheless they retain the desirable properties of not changing the communications graph and not introducing any extra communication events (assuming in the synchronous case that the timestamp "exchange" is built in to the message-passing implementation). **"Analyzing Consistency Properties for Fun and Profit",W. Golab, X. Li, and M. Shah.[7]**have motivated by the increasing popularity of eventually consistent key-value stores as a commercial service, It address two important problems related to the consistency properties in a history of operations on a read/write register (i.e., the start time, finish time, argument, and response of every operation). It presents efficient algorithms that compute these quantities. It believes that addressing these problems helps both key-value store providers and users adopt data consistency as an important aspect of key-value store offerings. **"Building a database on S3",MatthiasBrantner, Daniela Florescu, David Graf, Tim Kraska. [8]** Have proposed a great deal of hype about Amazon's simple storage service (S3). S3 provides infinite scalability and high availability at low cost. Currently, S3 is used mostly to store multi-media documents (videos, photos, audio) which are shared by a community of people and rarely updated. Web-based applications need high scalability and availability at low and predictable cost. Utility computing, however, is a viable candidate for many Web 2.0 and interactive applications. All of are aware that our approach to abandon strict consistency and DB-style transactions for the sake of scalability and availability is controversial and requires further discussion and evaluation. **"Consistency rationing in the cloud: pay only when it matters",TimKraska, Martin Hentschel, Gustavo Alonso, Donald Kossmann [9]** have discussed in Cloud storage solutions promise high scalability and low Cost Existing solutions, however, differ in the degree of consistency they provide. It has demonstrated the feasibility and potential of the ideas through extensive experiments on a first prototype implemented on Amazon's S3 and running the TPC-W benchmark. Our experiments indicate that the adaptive strategies presented in the paper result in a significant reduction in response time and costs including the cost penalties of inconsistencies. **"Quality-of-service for**

consistency of data geo-replication in cloud computing", Sergio Esteves, Joao Silva, and Luis Veiga. [10] Have discussed about the Cloud computing has recently emerged as a key technology to provide individuals and companies with access to remote computing and storage infrastructures. In order to achieve highly-available yet high-performing services, cloud data stores rely on data replication. However, providing replication brings with it the issue of consistency. Performance in HBase improves as the number of resources increases, for instance with more memory available, but it is not always trivial to scale further following that approach. Therefore, having ways of providing different levels of consistency to users regarding data in cloud environments translates into substantial traffic savings and potentially associated costs to service providers.

3. Methodology

The main objective of this project focuses on a key problem of using the replication technique in clouds is that it is very expensive to achieve strong consistency on a worldwide scale. In this paper, it first presents a novel consistency as a service (CaaS) model, which consists of a large data cloud and multiple small audit clouds. In the CaaS model, a data cloud is maintained by a CSP, and a group of users that constitute an audit cloud can verify whether the data cloud provides the promised level of consistency or not. It proposes a two-level auditing architecture, which only requires a loosely synchronized clock in the audit cloud. Then, we design algorithms to quantify the severity of violations with two metrics: the commonality of violations, and the staleness of the value of a read. Finally, It devise a heuristic auditing strategy (HAS) to reveal as many violations as possible.

3.1 Proposed System

In this paper, consistency as a service (CaaS) model and a two-level auditing structure to help users verify whether the cloud service provider (CSP) is providing the promised consistency, and to quantify the severity of the violations, if any. With the CaaS model, the users can assess the quality of cloud services and choose a right CSP among various candidates, e.g., and the least expensive one that still provides adequate consistency for the users' applications.

3.2 System Design

User Model View
This view represents the system from the user's perspective.
The analysis representation describes a usage scenario from the end-users perspective.

Structural Model view
In this model the data and functionality are arrived from inside the system.
This model view models the static structures.

Behavioral Model View It represents the dynamic of behavioral as parts of the system, depicting the interactions of collection between Cass Various structural elements described in the user model and structural model view.

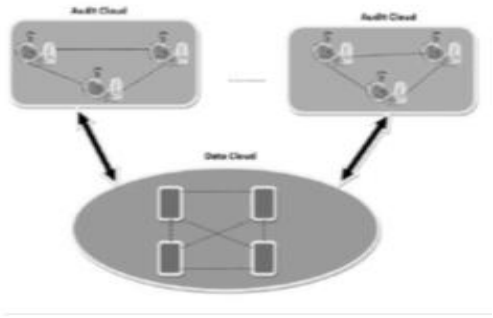


Figure 2: Behavioral Model

Implementation Model View
In this the structural and behavioral as parts of the system are represented as they are to be built.

Environmental Model View
In this the structural and behavioral aspects of the environment in which the system is to be implemented are represented.

3.3 Local consistency auditing:

Each user can perform *local auditing* independently with a local trace of operations; periodically, local auditing focuses on monotonic-read and read-your-write consistencies, which can be performed by a light-weight online algorithm.
Consistency model that allows it to automatically adjust the consistency levels for different semantic data.

3.4 Global consistency auditing.

An auditor is elected from the audit cloud to perform *global auditing* with a global trace of operations. In this case, all other users will send their UOTs to the auditor, Global auditing focuses on causal consistency, which is performed by constructing a directed graph. If the constructed graph is a

Directed acyclic graph (DAG), we claim that causal consistency is preserved.
Algorithm 1: Local consistency auditing
Step1: Initial UOT with
while issue an operation *op* **do**
Step2: if $op = W(a)$ **then**
record $W(a)$ in UOT
Step3: if $op = r(a)$ **then**
$W(b) \in$UOT is the last write
Step4: if $W(a) \rightarrow W(b)$ **then**
Step5: Read-your-write consistency is violated
$R(c) \in$UOT is the last read
Step6: if $W(a) \rightarrow W(c)$ **then**
Step7: Monotonic-read consistency is violated
record $r(a)$ in UOT
Algorithm 2 : Global consistency auditing
Step1: Each operation in the global trace is denoted by a vertex
for any two operations $op1$ and $op2$ **do**
Step2: if $op1 \rightarrow op2$ **then**
A time edge is added from $op1$ to $op2$
Step3: if $op1 = W(a)$, $op2 = R(a)$, and two operations come from different users **then**
A data edge is added from $op1$ to $op2$

Step4: if *op*1 = *W*(*a*), *op*2 = *W*(*b*), two operations come from different users, and *W*(*a*) is on the route from *W*(*b*) to

R(*b*) **then**
A causal edge is added from *op*1 to *op*2
Step5: Check whether the graph is a DAG by topological sorting

3.5 Data flow diagram

Data flow diagrams illustrate how data is processed by a system in terms of inputs and outputs. Data flow diagrams can be used to provide a clear representation of any business function. The technique starts with an overall picture of the business and continues by analyzing each of the functional areas of interest. This analysis can be carried out in precisely the level of detail required. The technique exploits a method called top-down expansion to conduct the analysis in a targeted way. As the name suggests, Data Flow Diagram (DFD) is an illustration that explicates the passage of information in a process. A DFD can be easily drawn using simple symbols. Additionally, complicated processes can be easily automated by creating DFDs using easy-to-use, free downloadable diagramming tools. A DFD is a model for constructing and analyzing information processes. DFD illustrates the flow of information in a process depending upon the inputs and outputs. A DFD can also be referred to as a Process Model. A DFD demonstrates business or technical process with the support of the outside data saved, plus the data flowing from the process to another and the end results.

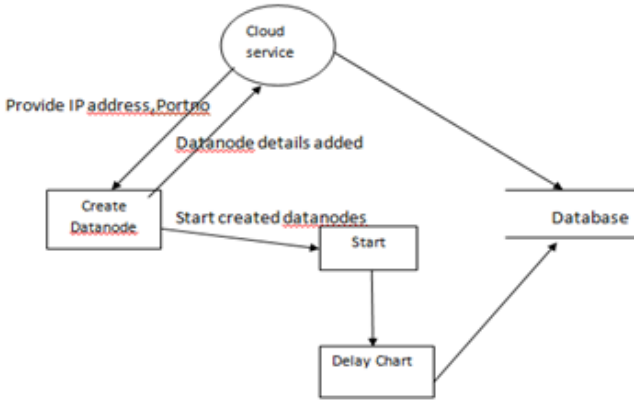


Figure 3: Data Flow Diagram: For CSP

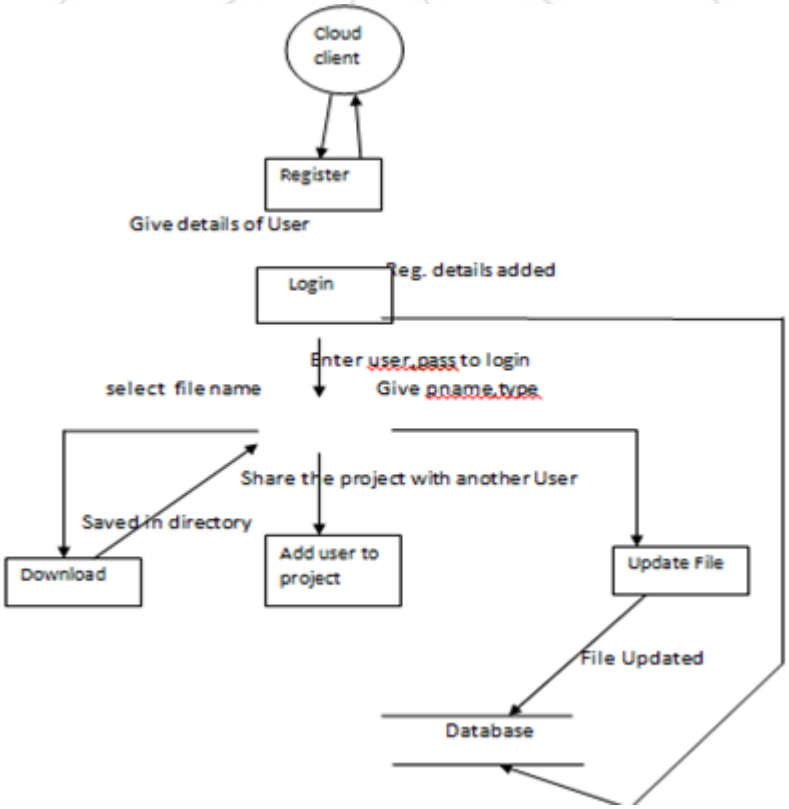
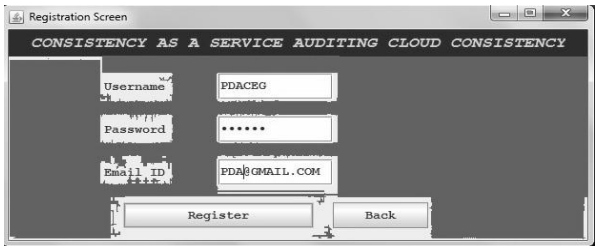
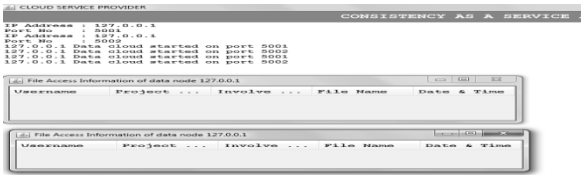


Figure 4: For cloud users

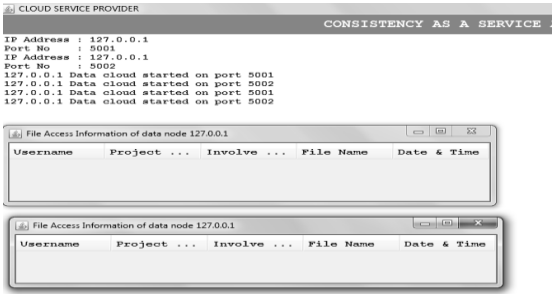
4. Results

Step1: Click on create data node to create a new data node





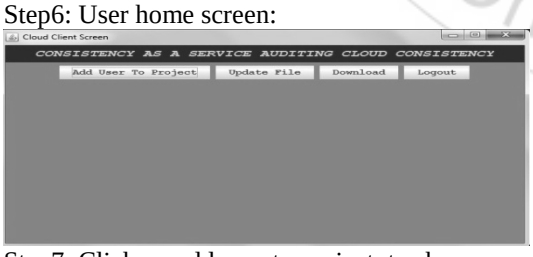
Step 2: Click on start button to start the all created data nodes



Step3: Run the auditor application, auditor home screen

Step4: Registering a new user:
Step5: User login screen:
Login as a user:

Step6: User home screen:



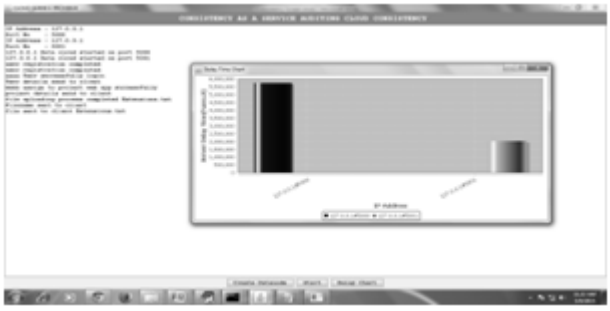
Step7: Click on add user to project, to share a project (work) with another user:

Step8: Click on update file to update a file to a user:

Select the project and select the consistency type then update a file

Step9: At server side we can see the delay chart for all the data nodes:

(The chart shows the time like how much time that each data node has taken to check the consistency)



5. Conclusion

In this paper, it is presented that consistency as a service (CaaS) model and a two-level auditing structure is used to help users to verify. Whether the cloud service provider (CSP) is providing the promised consistency, and to quantify the severity of the violations, if any. With the CaaS model, the users can assess the quality of cloud services and choose a right CSP among various candidates, e.g., the least expensive one that still provides adequate consistency for the users' applications. (For our future work, will conduct a thorough theoretical study of consistency models in cloud computing.)

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] "Data Consistency Properties and the Trade-offs in Commercial Cloud Storages:the Consumers Perspective" ,Hiroshi Wada, Alan Fekete, Kevin Lee, Anna Liu.

[3] "Pushing the CAP: strategies for consistency and availability," Computer, vol. 45, no. 2 , 2012.

[4] W. Lloyd, M. Freedman, M. Kaminsky, and D. Andersen, "Don't settle for eventual: scalable causal consistency for wide-area storage with COPS," in Proc. 2011 ACM SOSP.

[5] E. Anderson, X. Li, M. Shah, J. Tucek, and J. Wylie, "What consistency does your key-value store actually provide," in Proc. 2010 USENIX HotDep.

[6] C. Fidge, "Timestamps in message-passing systems that preserve the partial ordering," in Proc. 1988 ACSC.

[7] W. Golab, X. Li, and M. Shah, "Analyzing consistency properties for fun and profit," in Proc. 2011 ACM PODC.

[8] "Building a database on S3", Matthias Brantner, Daniela Florescu, David Graf, Tim Kraska.

[9] "Consistency rationing in the cloud: pay only when it matters",Tim Kraska ,Martin Hentschel, Gustavo Alonso, Donald Kossmann.

[10] "Quality-of-service for consistency of data geo-replication in cloud computing", Sergio Esteves, Joao Silva, and Luis Veiga.