IISRC International Conference on Industrial Scientific Research Engineering
International Industrial Scientific Research Corporation
www.worldconferences.org

Auditing Consistency of Cloud using Consistency as Service

NITHESH KUMAR K[1], P.SHAKEEL AHAMED[2], P.BABU[3]
[1]PG Scholar, Dept of CSE, Quba College of Engineering and Technology, Nellore, AP, India.
[2]Associate Professor, Dept of CSE, Quba College of Engineering and Technology, Nellore, AP, India.
[3]Associate Professor, Dept of CSE, Quba College of Engineering and Technology, Nellore, AP, India.

Abstract: Cloud storage services have become commercially popular due to their overwhelming advantages .To provide ubiquitous 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 first present 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. We propose 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, we devise a heuristic auditing strategy (HAS) to reveal as many violations as possible. Extensive experiments were performed using a combination of simulations and real cloud deployments to validate HAS.

Keywords: Service Level Agreement (SLA), NoSQL (Not Only SQL), Heuristic Auditing Strategy (HAS).

I. INTRODUCTION

Our model consist of large data cloud and 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, e.g., a document or a project. A service level agreement (SLA) will be engaged between the data cloud and the audit cloud, which will stipulate what level of consistency the data cloud should provide, and how much (monetary or otherwise) will be charged if the data cloud violates the SLA. The implementation of the data cloud is opaque to all users due to the virtualization technique. Thus, it is hard for the users to verify whether each replica in the data cloud is the latest one or not. We allow the users in the audit cloud to verify cloud consistency by analyzing a trace of interactive operations. Unlike their work, we do not require a global clock among all users for total ordering of operations. A loosely synchronized clock is suitable for our solution. The scope of this project is to upload and download a file from cloud. While providing cloud consistency, the following objectives are to be met: 1] Understanding the novel consistency as a service (CaaS) model provided by the cloud service provider. 2] The cloud computing solution should provide basic consistency as service.

II. LITERATURE SURVEY

A. Eventual Consistency: How Soon Is Eventual?

Over the last few years, Cloud storage systems and so-called NoSQL datastores have found widespread adoption. In contrast to traditional databases, these storage systems typically sacrifice consistency in favor of latency and availability as mandated by the CAP theorem, so that they only guarantee eventual consistency. Existing approaches to bench-mark these storage systems typically omit the consistency dimension or did not investigate eventuality of consistency guarantees. In this work we present a novel approach to benchmark staleness in distributed data stores and use the approach to evaluate Amazon's Simple Storage Service (S3).

B. Data Consistency Properties and the Trade offs in Commercial Cloud Storages: the Consumers' Perspective

A new class of data storage systems, called NoSQL (Not Only SQL), has emerged to complement traditional database systems, with rejection of general ACID transactions as one common feature. Different platforms, and indeed different primitives within one NoSQL platform, can offer various consistency properties, from Eventual Consistency to single-entity ACID. For the platform provider, weaker consistency should allow better availability, lower latency, and other benefits. This paper investigates what consumers observe of the consistency and performance properties of various offerings. We find that many platforms seem in practice to offer more consistency than they promise; we also find cases where the platform offers consumers a choice between stronger and weaker consistency, but there is no observed benefit from accepting weaker consistency properties.

**International Conference on Industrial Scientific Research Engineering Conference No.05,
April-2014, Pages:028-031**

C. Quality-of-Service for Consistency of Data Geo-replication in Cloud Computing

Today we are increasingly more dependent on critical data stored in cloud data centers across the world. To deliver high-availability and augmented performance, different replication schemes are used to maintain consistency among replicas. With classical consistency models, performance is necessarily degraded, and thus most highly-scalable cloud data centers sacrifice to some extent consistency in exchange of lower latencies to end-users. More so, those cloud systems blindly allow stale data to exist for some constant period of time and disregard the semantics and importance data might have, which undoubtedly can be used to gear consistency more wisely, combining stronger and weaker levels of consistency. To tackle this inherent and well-studied trade-off between availability and consistency, we propose the use of V FC3, a novel consistency model for replicated data across data centers with framework and library support to enforce increasing degrees of consistency for different types of data (based on their semantics). It targets cloud tabular data stores, offering rationalization of resources (especially bandwidth) and improvement of QoS (performance, latency and availability), by providing strong consistency where it matters most and relaxing on less critical classes or items of data.

D. Consistency Rationing In The Cloud: Pay Only When It Matters

Cloud storage solutions promise high scalability and low cost. Existing solutions, however, differ in the degree of consistency they provide. Our experience using such systems indicates that there is a non-trivial trade-off between cost, consistency and availability. High consistency implies high cost per transaction and, in some situations, reduced availability. Low consistency is cheaper but it might result in higher operational cost because of, e.g., overselling of products in a Web shop. In this paper, we present a new transaction paradigm, that not only allows designers to define the consistency guarantees on the data instead at the transaction level, but also allows to automatically switch consistency guarantees at runtime. We present a number of techniques that let the system dynamically adapt the consistency level by monitoring the data and/or gathering temporal statistics of the data. We demonstrate 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.

E. Analyzing Consistency Properties For Fun And Profit

Motivated by the increasing popularity of eventually consistent key-value stores as a commercial service, we 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). First, we consider how to detect a consistency violation as soon as one happens. To this end, we formulate a specification for online verification algorithms, and we present such algorithms for several well-known consistency properties. Second, we consider how to quantify the severity of the violations, if a history is found to contain consistency violations. We investigate two quantities: one is the staleness of the reads, and the other is the commonality of violations. For staleness, we further consider time-based staleness and operation-count-based staleness. We present efficient algorithms that compute these quantities. We believe that addressing these problems helps both key-value store providers and users adopt data consistency as an important aspect of key-value store offerings.

III. EXISTING SYSTEM

Although the existing schemes aim at providing integrity verification for different data storage systems, the problem of supporting both public audit ability and data dynamics has not been fully addressed. How to achieve a secure and efficient design to seamlessly integrate these two important components for data storage service remains an open challenging task in Cloud Computing.

Disadvantages Of Existing System:

- Although the infrastructures under the cloud are much more powerful and reliable than personal computing devices, they are still facing the broad range of both internal and external threats for data integrity.
- Second, there do exist various motivations for CSP to behave unfaithfully toward the cloud users regarding their outsourced data status.
- In particular, simply downloading all the data for its integrity verification is not a practical solution due to the expensiveness in I/O and transmission cost across the network. Besides, it is often insufficient to detect the data corruption only when accessing the data, as it does not give users correctness assurance for those un accessed data and might be too late to recover the data loss or damage.
- Encryption does not completely solve the problem of protecting data privacy against third-party auditing but just reduces it to the complex key management domain. Unauthorized data leakage still remains possible due to the potential exposure of decryption keys.

IV. PROPOSED SYSTEM

We propose a heuristic auditing strategy (HAS) which adds appropriate reads to reveal as many violations as possible. Our key contributions are as follows: 1) We present a novel 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) We propose a two-level auditing structure, which only requires a loosely synchronized clock

for ordering operations in an audit cloud. 3) We design algorithms to quantify the severity of violations with different metrics. 4) We devise a heuristic auditing strategy (HAS) to reveal as many violations as possible. Extensive experiments were performed using a combination of simulations and real cloud deployments to validate HAS.

Advantage:

- As a rising subject, cloud consistency is playing an increasingly important role in the decision support activity of every walk of life.
- Get Efficient Item set result based on the caas.

V. DFD Diagrams

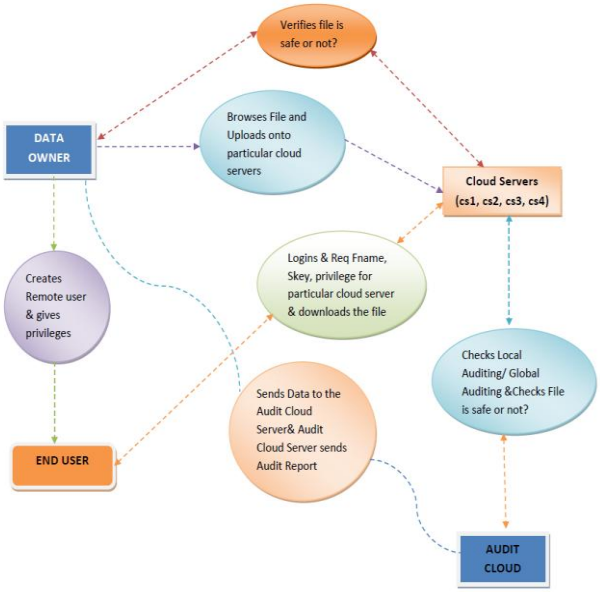

Fig.1.

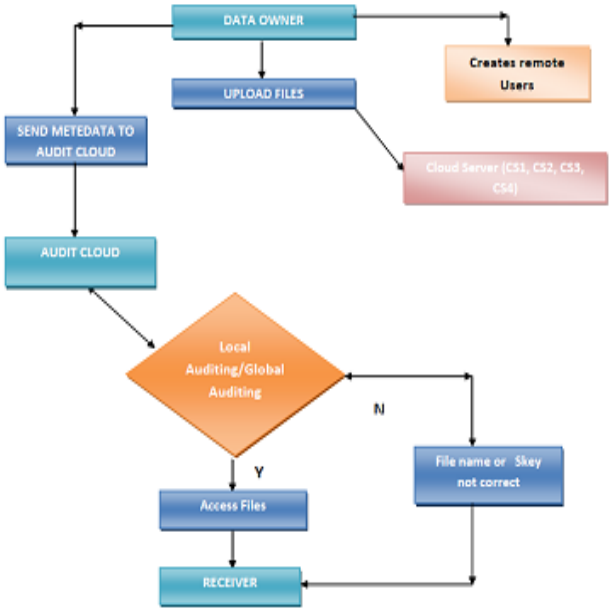

Fig.2. Flow chart.

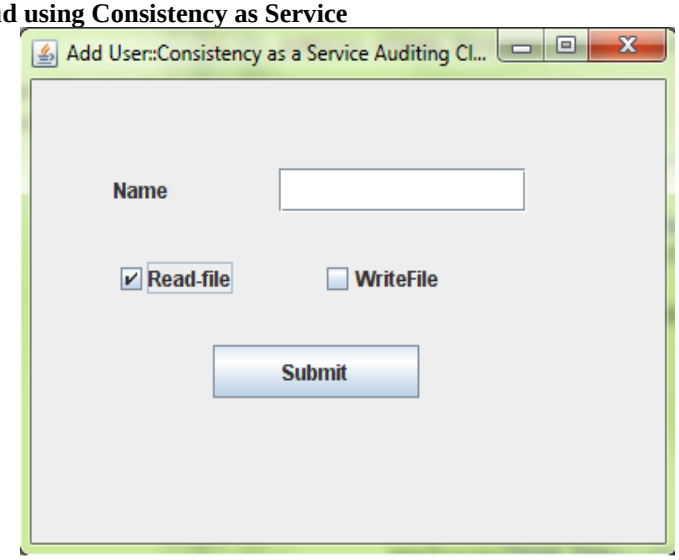

Fig.3.

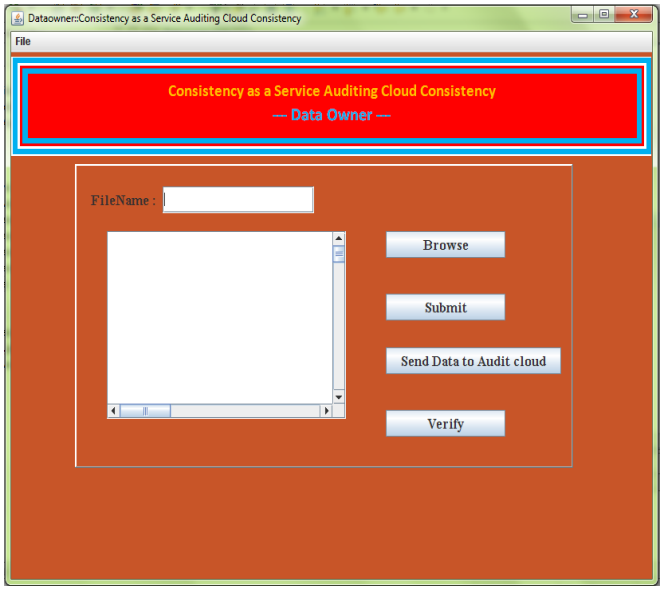

Fig.4.

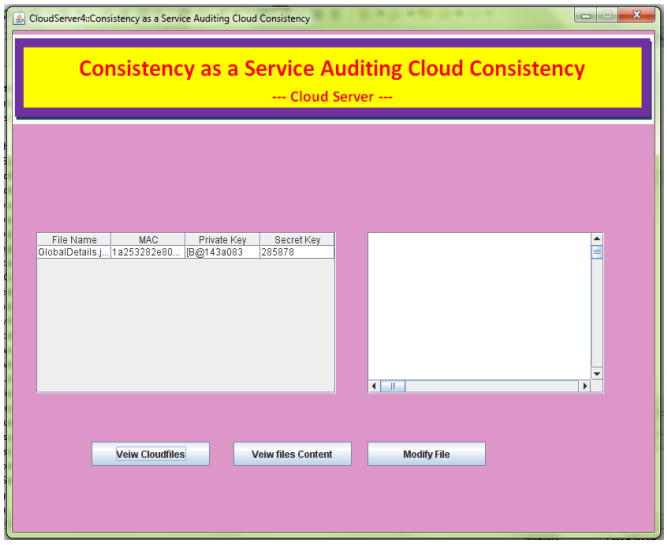

Fig.5.

Fig.6.

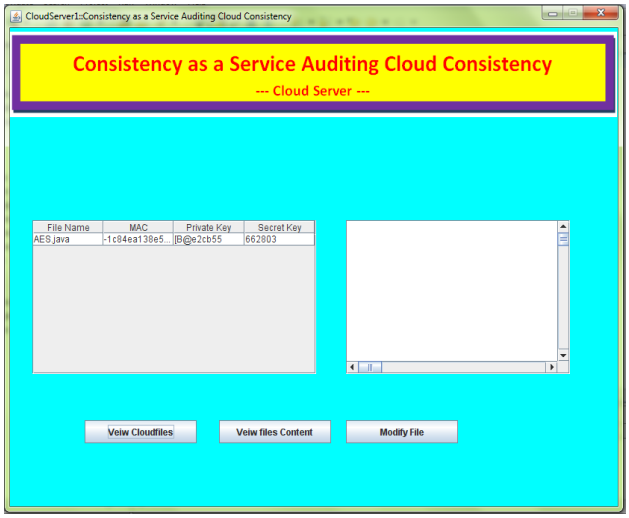

Fig.7.

VI. 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, "Towards robust distributed systems," in Proc. 2000 ACM PODC.

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

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

[6] 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.

[7] 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.

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

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

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

[11] W. Vogels, "Data access patterns in the Amazon.com technology platform," in Proc. 2007 LDB.

[12] "Eventually consistent," Commun. ACM, vol. 52, no. 1, 2009.

[13] M. Brantner, D. Florescu, D. Graf, D. Kossmann, and T. Kraska, "Building a database on S3," in Proc. 2008 ACM SIGMOD.

[14] T. Kraska, M. Hentschel, G. Alonso, and D. Kossmann, "Consistency rationing in the cloud: pay only when it matters," in Proc. 2009 VLDB.

[15] S. Esteves, J. Silva, and L. Veiga, "Quality-of-service for consistency of data geo-replication in cloud computing," Euro-Par 2012 Parallel Processing, vol. 7484, 2012.

[16] H. Wada, A. Fekete, L. Zhao, K. Lee, and A. Liu, "Data consistency properties and the trade-offs in commercial cloud storages: the consumers' perspective," in Proc. 2011 CIDR.

[17] D. Bermbach and S. Tai, "Eventual consistency: how soon is eventual?" in Proc. 2011 MW4SOC.