International Journal of Research (IJR)
e-ISSN: 2348-6848, p- ISSN: 2348-795X Volume 2, Issue 11, November 2015
Available at http://internationaljournalofresearch.org

Consistency as a Provision: Maintaining Cloud Consistency Using Auditing

K.Siva Rama Krishna#1& N.Madhava Rao #2

#1Student of M.Tech (CSE) and Dept. of CSE,MLEC College Of Science And Technology , Singarayakonda , A.P.
2Assoc. Professor, Dept. of CSE, MLEC College Of Science And Technology , Singarayakonda , A.P.

Abstract:

Cloud storage services are commercially more popular due to their amount of advantages. Cloud storage services with larger number of benefits have become commercially popular now- a- days. And they provide services like data storage services, and infrastructure management 24/7 through any device and from anywhere. The cloud service provider popularly known as CSP provides generally ubiquitous always-on-service by maintaining a single piece of data on different servers, which are geographically located in different places. The problem is that it is very expensive and failed to provide highly required consistency of service. Hence to overcome this issue, we going to propose a new fresh approach of service that is Consistency as as Service know as CaaS. The Consistency as a Service (CaaS) model concentrates on; in this we have large data cloud and small multiple audit clouds. Now firstly in the CaaS model, the main data cloud is created by a CSP, and a small number of group of users form an audit cloud part that can check whether the data cloud assures the guaranteed level of consistency that is whether it provides quality of service or not. To perform such operation on cloud we are going for two-level auditing strategy which makes use of loosely synchronized clock for calling operations in an audit cloud. Then perform global auditing by global trace of operations through randomly electing an auditor from an audit cloud. Finally, use a heuristic auditing strategy (HAS) to display as many violations as possible. Then randomly choosing an auditor from an audit cloud to perform global auditing operations i.e. to perform global trace of operations. And then finally, making use of Heuristic auding strategy (HAS), which display the possible violations.

Keywords--- Cloud storage; consistency as a service (CaaS); two-level auditing; and heuristic auditing strategy (HAS)

I. INTRODUCTION:Clouds computing has become more popular choice, because it has succeeded in giving guaranteed basic services like virtualized infrastructure system and providing data storage, etc. e.g. Amazon, SimpleDB are example of such systems. The customers or end users by making use of these services, become authorized users and able to access the data from anywhere and at any time using any device and getting confidence that the capital investment is going to less. The cloud service provider popularly known as CSP promising the users data is going to be available as 24/7, and they can access it efficiently. The CSP stores the different copies of data in a distributed fashion on different servers, which geographically present in different places. The main issue with distributing multiple copies of data called as replication technique is resultant into a very expensive process to provide strong consistency operation. In the coming days user is assured to see the latest updates about this service or operation. Many cloud service providers provide week consistency, we call such consistency as eventual consistency, where a user can read the data for particular time. Now-a-days stronger consistency assurance is getting importance. Consider the following figure. In the above figure data is stored in multiple copies on five cloud servers (CS1, CS2…, CS5), users specified in the figure share data through a cloud storage service. Here the cloud should provide casual consistency service, where a user Alice uploads a data on the cloud server CS4. Here the user update should be reflected in all the

International Journal of Research (IJR)
e-ISSN: 2348-6848, p- ISSN: 2348-795X Volume 2, Issue 11, November 2015
Available at http://internationaljournalofresearch.org

servers. If cloud service provider provides only eventual consistency then receiver user is going to receive the old version of data. Such a integrated design based on traditional version may not satisfy customer requirements.

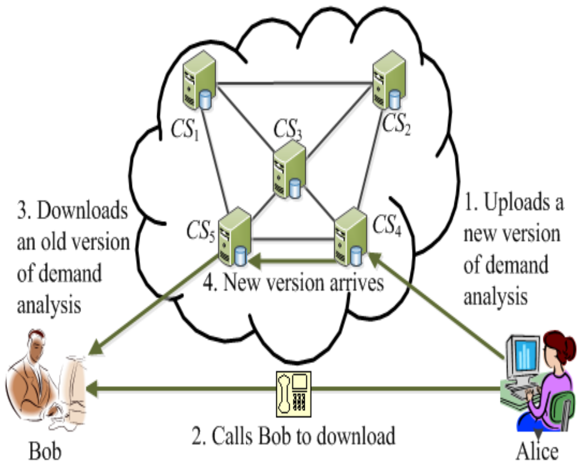

Fig. 1 Example to show casual consistency

Key Factors are as follows:

1) System Present CaaS model, which consist of data cloud and audit cloud

2) System suggests a two level auditing structure.

3) System design algorithms to measure the occurrences of violations with different metrics. 4) In this system devise HAS to find out as many as violations possible.

II. RELATED WORK A cloud is basically a major distributed system where each portion of data is copied on multiple globally distributed servers to attain high accessibility and high performance. Thus, we first check the consistency models in distributed systems. Ref. [10], as anticipated two consistency models: data-centric consistency and client-centric consistency. Datacentric consistency model consider the inner state of a storage system, that how updates stream through the system and what guarantees the system can supply with respect to updates. On the other hand, to a customer, it actually does not matter whether or not a storage system inside contains any old copies. As long as no old data is observed from the client's side, the customer is satisfied. Therefore, client-centric consistency model focuses on what exact customers want, with the aim of is how the customers view data updates. Their work also describes multiple levels of consistency in distributed systems, as of strict consistency to weak consistency. High consistency results in high cost and reduced availability. Firm consistency is never necessary in practice [11], and is even considered detrimental. In reality, by the CAP protocol [3],

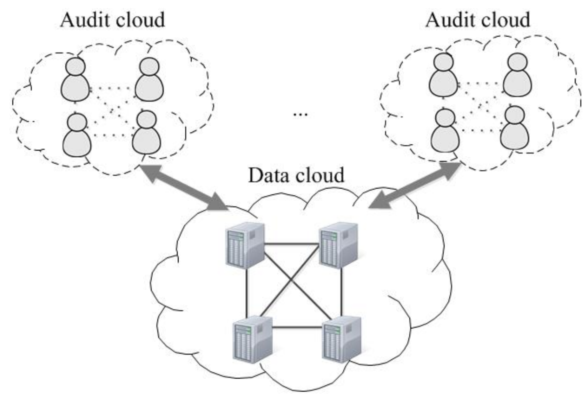

Fig.2 Consistency as a service model.

[4], many distributed systems forgo strict consistency for availability. Then, the system analyzes the work on attaining different levels of cloud. Investigated the consistency properties provided by commercial clouds and made several useful opinions [12]. Existing commercial clouds generally limit strong consistency promises to small datasets (Google's Megastore and Microsoft's SQL Data Services), or provide only eventual consistency (Amazon's simpleDB and Google's BigTable) [13]. The consistency requirements differ over time depending on tangible accessibility of the data, and the authors deliver techniques that make the system dynamically adjust to the consistency level by monitoring the state of the data. The proposed novel consistency model that allows it to automatically modify the consistency levels for altered semantic data [14]. Finally, we analyze the work on authenticating the levels of consistency provided by the CSPs from the user's point of view. Existing solutions can be categorized into trace-based verifications [7], [9] and benchmark-based verifications [15]- [18]. Trace-based verifications focus

on three consistency semantics: safety, regularity. A register is safe if a read that is not coexisting with any write returns the value of the most recent write, and a read that is coexisting with a write can return A register is regular if a read that is not coexisting with any write returns the value of the most contemporary write, and a read that is coexisting with a write returns either the value of the most contemporary write, or the value of the coexisting write. A register is atomic if every read returns the value of the most contemporary write. Misra [19] is the first to present an algorithm for confirming whether the suggestion on a read/write catalog is atomic. Following his work, Ref. [7] proposed offline algorithms for validating whether a key-value storage system has protection, reliability, and atomicity properties by assembling a directed graph. Ref. [9] offered an online verification algorithm by using the GK algorithm [20], and Used diverse metrics to enumerate the brutality of violations. The main weakness of the existing trace-based authentications is that a global clock is required among all users. Our solution belongs to trace-based authentications. However system emphasis on different consistency semantics in commercial cloud systems, where a loosely synchronized clock is proper for our explanation Benchmark-based authentications emphasis on benchmarking in a storage system. The results of validate our two-level auditing structure. Refer client centric benchmarking approach for understanding ultimate consistency in scattered key value storage systems. Amazon, Google, and Microsoft's contributions showed that, in Amazon S3, consistency was surrendered and only a weak consistency level known as, eventual consistency was attained.

III. PRELIMINARIES

In preliminaries section, first system illustrates the consistency as a service (CaaS) model. Then, it illustrates the structure of the user operation table (UOT), with which each user records his operations. Lastly, system makes available an overview of the two-level auditing structure and associated definitions.

A. Consistency as a Service (CaaS) Model

The CaaS model consists of a data cloud and multiple audit clouds. Data cloud is maintained by the cloud service provider (CSP), is a key-value data storage system where each part or piece of data is recognized by a unique key. The CSP replicates all of the data on multiple geographically distributed cloud servers to afford always-on services. An audit cloud consists of a group of users that assist on a job. Now assume that each user in the audit cloud is identified by a unique ID. The audit cloud and the data cloud will engage in a service level agreement (SLA), before outsourcing the job to the data cloud, Which specifies the promised level of consistency. The audit cloud verify whether the data cloud violates the SLA or not, and to enumerate the severity of violations. In this system, a two-level auditing model is implemented: each user records his operations in a user operation table (UOT), which is referred to as a local trace of operations. Local auditing can be carry out freely by each user with his own UOT; periodically, an auditor is designated from the audit cloud. In this, all other users will send their UOTs to the auditor, which will present global auditing with a global trace of operations. The system simply let each user turn into an auditor. The dotted line in the audit cloud shows that users are loosely connected. It implies that users will communicate to exchange messages after executing a set of reads or writes, rather than communicating instantly after executing each operation. Once two users finish communication, a causal relationship on their operations is established.

B. User Operation Table (UOT)

Each record in the UOT has three elements: operation, logical vector, and physical vector. User will record operation, his current logical vector and physical vector, while issuing an operation in his UOT. C. Overview of Two-Level Auditing Structure System examined several consistency models provided by profitable cloud systems. Following their work, we provide a two-level auditing structure for the CaaS model. At the first each user independently performs local auditing at his own with UOT. The following consistencies should be verified at this level Monotonic-read consistency. If a

International Journal of Research (IJR)
e-ISSN: 2348-6848, p- ISSN: 2348-795X Volume 2, Issue 11, November 2015
Available at http://internationaljournalofresearch.org

process reads the value of data, any successive reads on data by that process will always return that same value or a more recent value

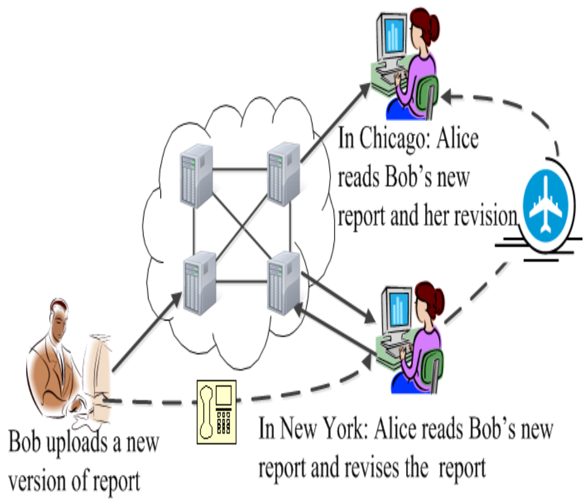

Fig.3. shows an application that has different consistency requirements.

In Fig. 3, after uploading a latest version of the report to the data cloud, Bob ask over Alice to download it. After the call, Bob's update and Alice's read are causally associated. Therefore, causal consistency needs that Alice must read Bob's new report. Read-your-write consistency. Effect of a write by a process on data K will always be seen by a successive read on data K by the same process. Causal consistency. Writes that are causally related must be seen by all processes in the similar order. Simultaneous writes may be seen in a different order on different machines

IV. VERIFICATION OF CONSISTENCY PROPERTIES

In this, systems first afford the algorithms for the two level auditing structure for the CaaS model, and then analyze their success. Finally, system demonstrates how to perform a trash collection on UOTs to save space. As the accesses of data with different keys are independent of each other, a user can group operations by key and then verify whether each group satisfies the promised level of consistency. After that, system reduces read operations with R (a) and writes operations with W (a).

A. Local Consistency Auditing Algorithm

Initial UOT with ∅
while issue an operation *op* **do**
if $op = W(a)$ **then**
record $W(a)$ in UOT
if $op = r(a)$ **then**
$W(b) \in$UOT is the last write
if $W(a) \rightarrow W(b)$ **then**
Read-your-write consistency is violated
$R(c) \in$UOT is the last read
if $W(a) \rightarrow W(c)$ **then**
Monotonic-read consistency is violated
record $r(a)$ in UOT

R (a) – Users Current Read

W (a) – Current reads dictating write

R (c) – Last Read in UOT

W (c) – Last Read in UOT's dictating write

W (b) – Last write in UOT

This is an online algorithm. In this each user will record all of his operations in his UOT. User will perform local consistency auditing independently.

B.Global Consistency Auditing

Algorithm 2 Global consistency auditing Every operation in the global trace is represented by a vertex

Let any two 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 two operations come from different users Then A data edge is constructed from op1 to op

If op1 = W (a), op2 = 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 op1 to op2 Check whether the graph is a DAG by topological sorting.

This is an offline algorithm (Alg. 2). An auditor will be chosen periodically from the audit cloud to perform global consistency auditing. All other users will submit their UOTs to the auditor for getting a global trace of operations. After performing global auditing, the auditor will send audit results as well as its vectors to all other.

C. Effectiveness

Effectiveness of the local consistency auditing algorithm is easy to demonstrate. For monotonic-read consistency, a user is needed to read either the same value or a newer value. Hence, if the dictating write of a new read happens before the dictating write of the last read, then system say that monotonic read consistency is violated. In case of read-your-write consistency, the user is needed to read his latest write. Hence, if the dictating write of a new read happens before his last write, system can say that read-your-write consistency is violated. For causal consistency, system should prove that:

(1) There is an violation if the constructed graph is not a DAG.

(2) There is no violation if the graph is DAG.

D. Garbage Collection

Each user should keep all operations in his UOT in the process of auditing, exclusive of intercession; the size of the UOT would grow without bound. Also, the communication cost for transferring the UOT to the auditor will be too much. So, system provides a garbage collection system which can delete unnecessary records, which will preserve the efficiency of auditing. In local consistency auditing, suppose dictating write of a new read does not exist in the UOT and the dictating write is issued by the user, the user can say that he has failed to read his last updates, and asserts that read-your-write consistency is violated. Suppose the dictating write of this read happens before the dictating write of his last read recorded in the UOT, the user can say that he has read an old value, and asserts that monotonic-read consistency is violated. Let the dictating write of a new read does not present in the user's UOT and the dictating write comes from other users, then a violation will be exposed by the auditor. In global consistency auditing, if a read that does not have a dictating write, then the auditor say that the value of this read is too stale, and state that causal consistency is violated.

Summary.

HAS can detect nearly all of the violations when the inception value and interval length are chosen accurately; Random can perceive only about 60% of destructions. Although HAS involves the auditing cloud to dispute more auditing reads, the grossed profit is still higher than Random. Specifically, as the parameters inception value and interval length reduce, HAS works better.

VII. EVALUATION

In this unit, system unite HAS with a random strategy, denoted as Random. To confirm the efficiency of HAS, system conduct tests on synthetic as well as real violation traces.

A. Synthetic Violation Traces

System review the parameters used in the artificial violation races in Table II. In the random strategy, system erratically choose [1, l] auditing reads in each recess, where l is the length of an recess. To obtain the synthetic violation traces, physical time is divided into 2,000 time slices. We accept that once a data cloud activates to violate the assured consistency, this violation will continue for several time slices, rather than ending instantaneously. In the recreation, the period of each violation d is set to 3-10 time slices.

Consider that the audit cloud can earn $5 from the data cloud once a consistency violation is detected; the audit cloud will be charged $0.1 for an auditing read task. Fig. 8 shows the contrast results of the earned profit P. From we know that HAS typically earns a higher profit than Random. Finally, HAS will produce higher earned profit as the parameters α and l decrease. produce higher earned profit as the parameters α and l decrease.

B. Real Violation Traces

To check the productivity of HAS, system collect traces from two real clouds. We use network time protocol (NTP) to coordinate time amongst all cases. We know that the proportion of exposed destructions reduces as l rises, in terms of both HAS as well as Random. However, the change of l's value has less impression on HAS than Random. We know that the proportion of exposed destructions drops as α rises or k falls. However, these factors have slight effects on the proportion of exposed destructions. We know the percentage of revealed violations decreases as α increases.

VIII. CONCLUSION

In this paper, The presented system is a consistency as a service (CaaS) model and a two-level auditing scheme to help users validate whether the cloud service provider (CSP) is providing the promised consistency, and to enumerate the occurrences of the violations. The CaaS model used in the system helps the users can assess the superiority of cloud services and decide a right CSP among various services. For example the less costly one that still provides satisfactory consistency for the users' applications.

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 VLDB.

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

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

[15] 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. [16] D. Bermbach and S. Tai, "Eventual consistency: how soon is eventual?" in Proc. 2011 MW4SOC.

[17] M. Rahman, W. Golab, A. AuYoung, K. Keeton, and J. Wylie, "Toward a principled framework for benchmarking consistency," in Proc. 2012Workshop on HotDep.

[18] D. Kossmann, T. Kraska, and S. Loesing, "An evaluation of alternative architectures for transaction processing in the cloud," in Proc. 2010 ACM SIGMOD.

[19] J. Misra, "Axioms for memory access in asynchronous hardware systems," ACM Trans. Programming Languages and Systems, vol. 8, no. 1, 1986.

[20] P. Gibbons and E. Korach, "Testing shared memories," SIAM J. Computing, vol. 26, no. 4, 1997.

Student Profile:

K.SIVA RAMA KRISHNA was born in andhra pradesh,india .i am recevied M.SC degree in 2012 from ANU guntur,guntur(DT). i am pursuing M.Tech degree in CSE from malineni lakshamaiah engineering college,singarayakonda.

Guide Profile:

N.MADHAVA RAO

Working as Associate Professor in MLEC CollegeSingarayakonda.M.Tech PG Received in the year of 2010. B.Tech Degree Received In the year of 1994.