Volume No:1, Issue No:4 (September-2015) ISSN No : 2454-423X (Online)

International Journal of Research in Advanced Computer Science Engineering

A Peer Reviewed Open Access International Journal

www.ijracse.com

Consistency as a Service: Auditing Cloud Consistency

Challa Haritha

Dept of Computer Science Engineering,
SKR College of Engineering and Technology,
Nh-5, Kondurusatram, Manubolu, Spsr Nellore,Ap.

Nagala Venkatadri

Associate Professor,
Dept of CSE & It,
SKR College of Engineering and Technology,
Nh-5, Kondurusatram, Manubolu, Spsr Nellore,Ap.

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 realcloud deployments to validate HAS.

EXISTING SYSTEM:

» By using the cloud storage services, the customers can access data stored in a cloud anytime and anywhere using any device, without caring about a large amount of capital investment when deploying the underlying hardware infrastructures.
» The cloud service provider (CSP) stores data replicas on multiple geographically distributed servers.
» Where a user can read stale data for a period of time.

The domain name system (DNS) is one of the most popular applications that implement eventual consistency. Updates to a name will not be visible immediately, but all clients are ensured to see them eventually.

DISADVANTAGES OF EXISTING SYSTEM:

» The replication technique in clouds is that it is very expensive to achieve strong consistency.
» Hard to verify replica in the data cloud is the latest one or not.

PROPOSED SYSTEM:

» In this paper, we presented a 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, the least expensive one that still provides adequate consistency for the users' applications.

ADVANTAGES OF PROPOSED SYSTEM:

» Do not require a global clock among all users for total ordering of operations.
» The users can assess the quality of cloud services.
» choose a right CSP
» Among various candidates, e.g, the least expensive one that still provides adequate consistency for the users' applications.

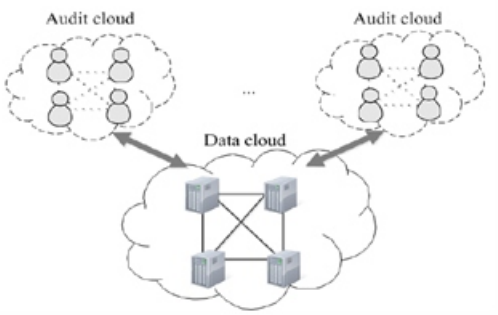
Volume No:1, Issue No:4 (September-2015) ISSN No : 2454-423X (Online)

International Journal of Research in Advanced Computer Science Engineering

A Peer Reviewed Open Access International Journal

www.ijracse.com

SYSTEM ARCHITECTURE:



MODULES :

1.System Module
2.User operation table
3.Local Consistency Auditing
4.Global Consistency Auditing

MODULES DESCRIPTION:

1.System Module:

•In the first module, we develop the System Module with User Module, Admin Module, and Auditor Module.
•In user module, user should register their details and get the secret key for login and user can upload the file regarding the auditing. In user module, the user uploaded files can be stored in cloud database. Auditor can view the file from the database it can be much secured.
•In admin module admin can view all the user details; user uploads details, and TPA activities regarding the auditing strategy.
•In auditor module, auditor can do the auditing based on the heuristic auditing strategy. It relates with document verification. Auditor can check the auditing file he can reject or accept the file he can revise the report and check whether it's good or bad. And auditor can give revision report like accept or waiting. If status in accept means user can view the file else status is waiting means user cant view the file.

2.User Operation Table:

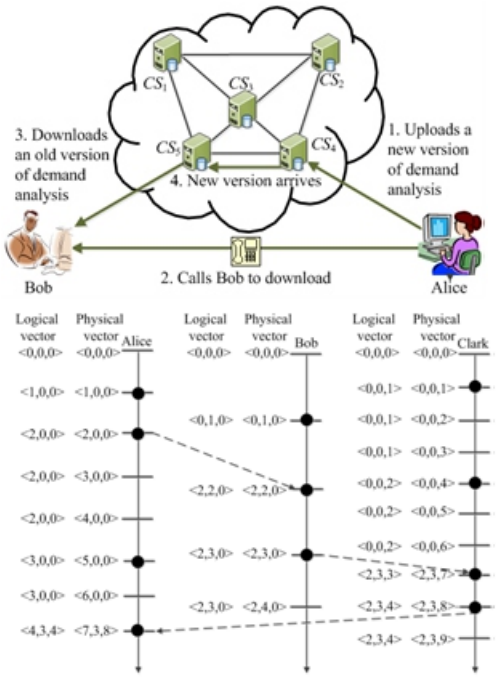
Each user maintains a UOT for recording local operations. Each record in the UOT is described by three elements: operation, logical vector, and physical vector. While issuing an operation, a user will record this operation, as well as his current logical vector and physical vector, in his UOT. Each user will maintain a logical vector and a physical vector to track the logical and physical time when an operation happens, resepectively.

3.Local Consistency Auditing:

Local consistency auditing is an online algorithm. In this module, each user will record all of his operations in his UOT. While issuing a read operation, the user will perform local consistency auditing independently.

4.Global Consistency Auditing:

Global consistency auditing is an offline algorithm. Periodically, an auditor will be elected from the audit cloud to perform global consistency auditing. In this case, 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. Given the auditor's vectors, each user will know other users' latest clocks up to global auditing.



The logical vector is updated via the vector clocks algorithm [8]. The physical vector is updated in the same way as the logical vector, except that the user's physical clock keeps increasing as time passes, no matter whether an event (read/write/send message/receive message) happens or not. The update process is as follows: All clocks are initialized with zero (for two vectors); The user increases his own physical clock in the physical vector continuously, and increases his own logical clock in the logical vector by one only when an event happens; Two vectors will be sent along with the message being sent.

Volume No:1, Issue No:4 (September-2015) ISSN No : 2454-423X (Online)

International Journal of Research in Advanced Computer Science Engineering

A Peer Reviewed Open Access International Journal

www.ijracse.com

When a user receives a message, he updates each element in his vector with the maximum of the value in his own vector and the value in the received vector (for two vectors). To illustrate, let us suppose that there are three users in the audit cloud, Alice, Bob, and Clark, where IDAlice < IDBob < IDClark. Each user may update the vectors as shown in Fig. 3.

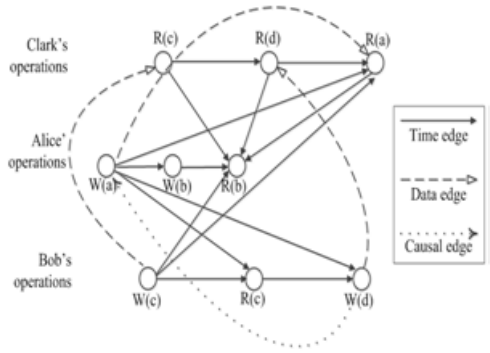
If the first event for Alice is W(K, a), the first record in Alice's UOT is [W(K, a),< 1, 0, 0 >,< 1, 0, 0 >]. This means that Alice writes value a to data identified by key K when both her physical and logical clocks are 1. Furthermore, when this event happens, she has no information about other users' clocks, which are thus set with the initial value 0. Note that, since there is no global time in the audit cloud, the number of clock ticks in each user's physical clock may be different, e.g., in Fig. 3, when Alice's physical clock passed seven clock ticks, Bob's physical clock passed only four ticks.

C. Overview of Two-Level Auditing StructureVogels [12] investigated several consistency models provided by commercial cloud systems. Following their work, we provide a two-level auditing structure for the CaaS model. At the first level, each user independently performs local auditing with his own UOT. The following consistencies (also referred to as local consistencies) should be verified at this level:Monotonic-read consistency. If a process reads the value of data K, any successive reads on data K by that process will

.

Algorithm 1 Local consistency auditing
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) UOT is the last write
if W(a) → W(b) then
Read-your-write consistency is violated
R(c) UOT is the last read
if W(a) → W(c) then
Monotonic-read consistency is violated
record r(a) in UOT
always return that same value or a more recent value.
Read-your-write consistency. The 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.

Intuitively, monotonic-read consistency requires that a user must read either a newer value or the same value, and readyour- write consistency requires that a user always reads his latest updates. To illustrate, let us consider the example in Fig.4. 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. 4, 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. At the second level, an auditor can perform global auditing after obtaining a global trace of all users' operations. At this level, the following consistency (also referred to as global consistency in this paper) should be verified:



VERIFICATION OF CONSISTENCY PROPERTIES:

In this section, we first provide the algorithms for the two level auditing structure for the CaaS model, and then analyze their effectiveness. Finally, we illustrate how to perform a garbage collection on UOTs to save space. Since 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. In the remainder of this paper, we abbreviate read operations with R(a) and write operations with W(a).

Volume No:1, Issue No:4 (September-2015) ISSN No : 2454-423X (Online)

International Journal of Research in Advanced Computer Science Engineering

A Peer Reviewed Open Access International Journal

www.ijracse.com

A. Local Consistency Auditing:

Local consistency auditing is an online algorithm (Alg. 1). In Alg. 1, each user will record all of his operations in his UOT. While issuing a read operation, the user will perform local consistency auditing independently.Let R(a) denote a user's current read whose dictating write is W(a), W(b) denote the last write in the UOT, and R(c) denote the last read in the UOT whose dictating write is W(c). Read-your-write consistency is violated if W(a) happens before W(b), and monotonic-read consistency is violated if W(a) happens before W(c). Note that, from the value of a read, we can know the logical vector and physical vector of its dictating write. Therefore, we can order the dictating writes by their logical vectors.

B. Global Consistency Auditing:

Global consistency auditing is an offline algorithm (Alg. 2). Periodically, an auditor will be elected from the audit cloud to perform global consistency auditing. In this case, 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 4Let LV (ei)j denote user j's logical clock in LV (ei). LV (e1) < LV (e2) if j[LV (e1)j ≤ LV (e2)j] j[LV (e1)j < LV (e2)j].

Algorithm 2 Global consistency auditing

Each operation in the global trace is denoted by a vertex

for 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 added from op1 to op2

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

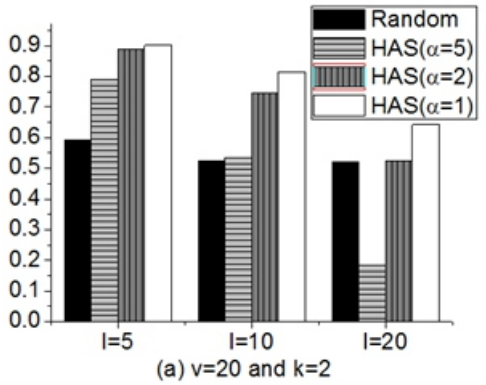
Fig. 5. Sample graph constructed with Alg. 2. users. Given the auditor's vectors, each user will know other users' latest clocks up to global auditing. Inspired by the solution in [7], we verify consistency by constructing a directed graph based on the global trace. We claim that causal consistency is preserved if and only if the constructed graph is a directed acyclic graph (DAG). In Alg.

2, each operation is denoted by a vertex. Then, three kinds of directed edges are added by the following rules:

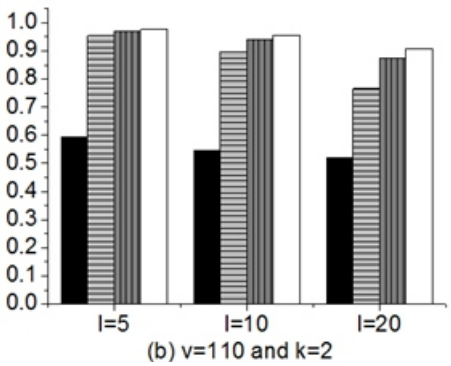
1) Time edge. For operation op1 and op2, if op1 → op2, then a directed edge is added from op1 to op2.

2) Data edge. For operations R(a) and W(a) that come from different users, a directed edge is added fromW(a) to R(a).

3) Causal edge. For operations W(a) and W(b) that come from different users, if W(a) is on the route from W(b) to R(b), then a directed edge is added from W(a) to W(b). Take the sample UOTs in Table I as an example. The graph constructed with Alg. 2 is shown in Fig. 5. This graph is not a DAG. From Table I, we know that W(a) → W(d), as LV (W(a)) < LV(W(d)). Ideally, a user should first read the value of a and then d. However, user Clark first reads the value of d and then a, violating causal consistency. To determine whether a directed graph is a DAG or not, we can perform topological sorting [25] on the graph. Any DAG has at least one topological ordering, and the time complexityof topological sorting is O(V +E), where V is the number of vertexes and E is the number of edges in the graph. To reduce the running time of topological sorting, we can modify Alg.EVALUATION In this section, we compare HAS with a random strategy, denoted as Random. To verify the ffectiveness of HAS, we conduct experiments on both synthetic and real violation traces. Our experiments are conducted with MATLAB R2010a running on a local machine, with an Intel Core 2 Duo E8400 3.0 GHz CPU and 8 GB Linux RAM.



(a) v=20 and k=2



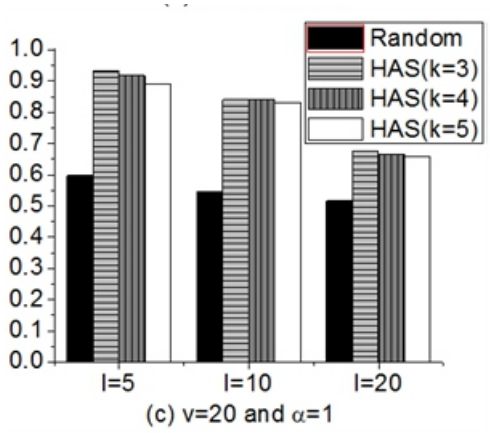
(b) v=110 and k=2

Volume No:1, Issue No:4 (September-2015) ISSN No : 2454-423X (Online)

International Journal of Research in Advanced Computer Science Engineering

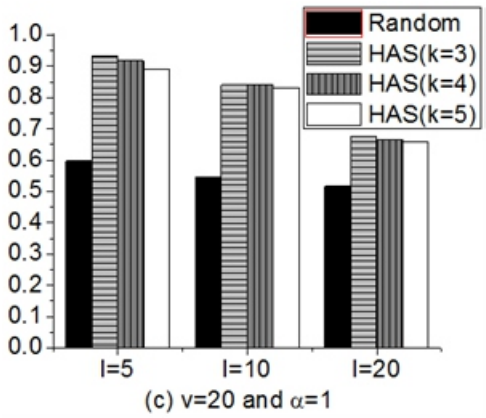
A Peer Reviewed Open Access International Journal

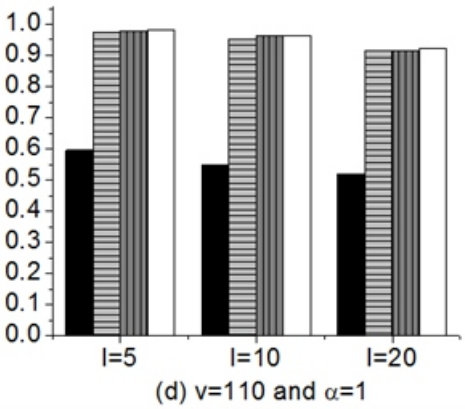
www.ijracse.com

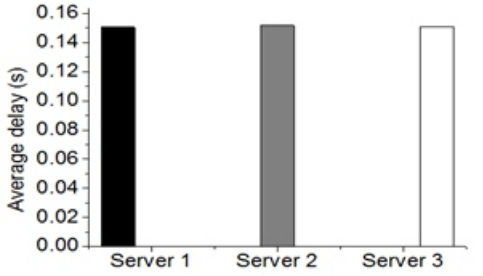


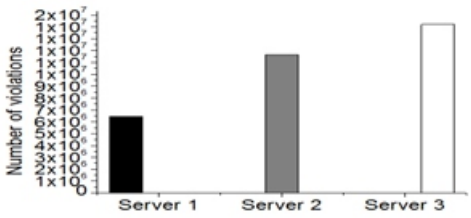
Synthetic Violation Traces:

We summarize the parameters used in the synthetic violation traces in Table II. In the random strategy, we randomly choose [1, l] auditing reads in each interval, where l is the length of an interval. To obtain the synthetic violation traces, physical time is divided into 2,000 time slices.









CONCLUSION:

In this paper, we presented a 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 various candidates, e.g, the least expensive one that still provides adequate consistency for the users' applications. For our future work, we 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] 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.

Volume No:1, Issue No:4 (September-2015) ISSN No : 2454-423X (Online)

International Journal of Research in Advanced Computer Science Engineering

A Peer Reviewed Open Access International Journal

www.ijracse.com

[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] 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.

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

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

[20] L. Lamport, "On interprocess communication," Distributed Computing, vol. 1, no. 2, 1986.

[21] A. Aiyer, L. Alvisi, and R. Bazzi, "On the availability of non strict quorum systems," Distributed Computing, vol. 3724, 2005.

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

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

[24] 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.

[25] T. Gormen, C. Leiserson, R. Rivest, and C. Stein, Introduction to Algorithms. MIT Press, 1990.

About Authors:

1.Challa Haritha: She Was Born In Utukuru [V], Sydapuram [M], S.P.S.R.Nellore [Dt], Andhra Pradesh, India. She Received The B.Tech Degree In Computer Science & Engineering From Jnt University, Anantapur In 2012 And Pursuing M.Tech Degree In Computer Science & Engineering From Jnt University, Anantapur. She Completed Her B.Tech Degree In Jb Women`S Engineering College, Renugunta Road, Tirupati [T], Chithoor [Dt], Andhra Pradesh And M.Tech Degree In Skr College Of Engineering & Technology, Konduru Satram [V], Manubolu [M], S.P.S.R.Nellore [Dt], Andhra Pradesh, India.

Volume No:1, Issue No:4 (September-2015) ISSN No : 2454-423X (Online)

International Journal of Research in Advanced Computer Science Engineering

A Peer Reviewed Open Access International Journal

www.ijracse.com

2.Mr. Nagala Venkatadri He Was Born In Andhrapradesh, India. He Received The B.Tech Degree From Jnt University, Anantapur And M.Tech Degree From Jnt University, Anantapur. He Has 10 Years Experience In The Field Of Associate Professor. He Had Working As Associate Professor In Dept. Of Computer Science & Engineering And Software Engineering In Skr College Of Engineering & Technology, Konduru Satram [V], Manubolu [M], S.P.S.R Nellore [Dt], Andhra Pradesh, India. He Is Presently Pursuing His Doctorate In Dept. Of Computer Science & Engineering.