ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 4, Issue 3, pp: (153-158), Month: July - September 2016, Available at: www.researchpublish.com

Distributed Gossip Protocol for Inconsistency Avoidance in Cloud Environment

Mumthas B

M.G University, Mount Zion College of Engineering, Pathanamthitta, India

Abstract: **Today's cloud storage service become popular. In cloud environment , cloud service provider maintains multiple number of copies of data on multiple distributed servers. The main problem of using replication technique, achieving strong consistency is difficult. So we propose a novel consistency as a service model which contains a large data cloud maintained by cloud service provider and group of users that constitute audit cloud. Audit cloud verify that data cloud provides maximum level of consistency. Between data cloud and audit cloud there will be a service level agreement engaged. For verify consistency properties two level auditing structure is used here. For this it only required a loosely synchronized clock. To avoid inconsistency in the cloud environment, gossip protocol is used. Gossip is style of computer- to -computer communication protocol. Modern distributed system use gossip protocol to solve their problems that might be difficult to solve in other ways.**

Keywords: **Cloud Storage, Gossip Protocol, SLA.**

I. INTRODUCTION

Cloud computing is nothing it means storing of data and application over the network. because it promises high scalability, elasticity, availability at low cost, cloud computing has become very popular. Cloud computing provides the most important service called cloud storage service, which deliver the data storage as a service, including network attached storage and database-like services. Amazon simpleDB, Microsoft Azure storage and so on are some of the examples. By using the cloud storage services, the customers can access data stored in a cloud anytime and anywhere using any hardware devices without any need of the capital investment. To provide the always on access over world wide scale, the cloud service provider (CSP) maintains the replicas of data on multiple servers. The main problem of using replication technique, achieving strong consistency is difficult. for high availability and performance, many cloud service providers provide eventual consistency, week consistency, ie, users can always read only old version. DNS is the One of the most popular application that provide eventual consistency.

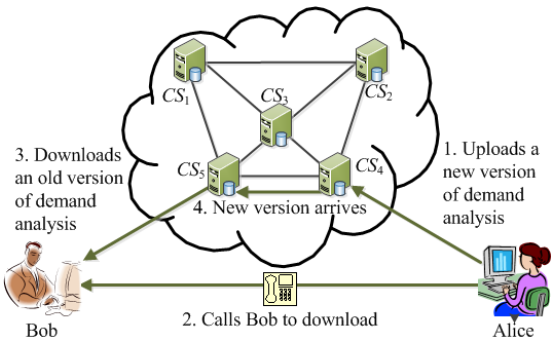

Fig.1: An application that requires causal consistency

Now a days many commercial applications use cloud storage services due to their submerged advantages. Consistency is a key requirement in cloud environment. Let's take an example, suppose that Alice and Bob are working on a project

together using a cloud storage service. That cloud containing five cloud servers, CS1,….,CS5. The data is replicated to this five cloud servers. Bob calls Alice to download the latest update of requirement analysis from CS_4, after updating it. Here, there exist a causal relationship between Bob's update and Alice's read after Bob calls Alice. So Alice can able to download the latest version of requirement analysis from the cloud server CS5. That means the data cloud should provide causal consistency, which ensures that Bob's update is replicated to all of the cloud servers before Alice read the update. If the cloud provide only eventual consistency, then Alice is allowed to access an old version from CS5. From this example, if the data cloud provides only eventual consistency, that not satisfy the requirements of customers. So in cloud environment need different consistencies for different applications.

In cloud environment , cloud service provider maintains multiple number of copies of data on multiple distributed servers. The main problem of using replication technique, achieving strong consistency is difficult. So we propose a novel consistency as a service model which contains a large data cloud maintained by cloud service provider and group of users that constitute audit cloud. Audit cloud verify that data cloud provides maximum level of consistency. Between data cloud and audit cloud there will be a service level agreement engaged. For verify consistency properties two level auditing structure is used here. For this it only required a loosely synchronized clock. To avoid inconsistency in the cloud environment, gossip protocol is used. Gossip is style of computer- to -computer communication protocol. Modern distributed system use gossip protocol to solve their problems that might be difficult to solve in other ways.

II. 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 integrate these two important components for data storage service remains an open challenging task in Cloud Computing.

- 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.

III. PROBLEM DEFINITION

Optimal electric vehicle charging is our problem and it can be defined as follows. We are given a fleet of EVs within a distribution grid and each EV must receive a specific amount of energy satisfying its daily demand. EV users are also interested in finding an energy optimal route with appropriate charging stops while not spending unnecessary time at the charging station which will allow them to extend the driving range. For EV fleets this situation is even more critical. Our aim is to supply power to each of the charging station from the power grid and hence to satisfy the energy demand of individual users by minimizing the financial costs associated with power supply.

IV. PROPOSED SYSTEM

In cloud storage, consistency not only determines correctness but also the actual cost per transaction. In this paper, we present a novel consistency as a service (CaaS) model for this situation. The CaaS model consists of 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, 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.

ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 4, Issue 3, pp: (153-158), Month: July - September 2016, Available at: www.researchpublish.com

The main aim of our proposed system is to avoiding inconsistency arises in the cloud environment. The best solution is NP hard problem. In NP hard problem, each and every node in the network communicate with each other and send their data or status. Suppose there are 10 nodes, each node send their data or status to other 9 nodes. So complexity become 10^{10}, it is not feasible.

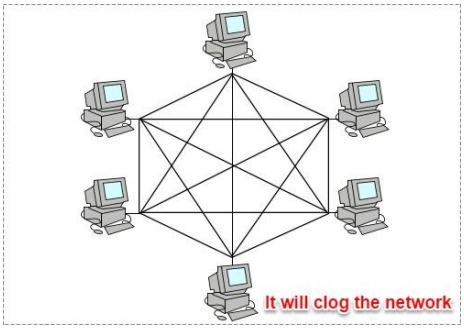

Fig.2: NP hard problem

So we introduce an another technique called gossip based update. Gossip is a style of computer to computer communication protocol inspired by the form of gossip seen in social network. Modern distributed system often use gossip protocol to solve problems that might be difficult to solve in other ways. The concept of the gossip communication can be illustrated by the analogy of office workers spreading rumours.

A. Consistency as a Service Model:

As shown in Fig. 3, the CaaS model consists of a data cloud and multiple audit clouds. The data cloud, maintained by the cloud service provider (CSP), is a key-value data storage system, where each piece of data is identified by a unique key. To provide always-on services, the CSP replicates all of the data on multiple geographically distributed cloud servers. An audit cloud consists of a group of users that cooperate on a job, e.g., a document or a program. We assume that each user in the audit cloud is identified by a unique ID. Before outsourcing the job to the data cloud, the audit cloud and the data cloud will engage in a service level agreement (SLA), which stipulates the promised level of consistency that should be provided by the data cloud. The audit cloud exists to verify whether the data cloud violates the SLA or not, and to quantify the severity of violations.

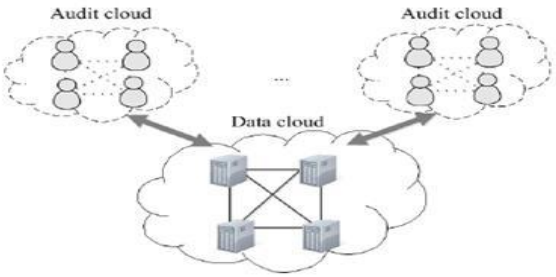

Fig.3: consistency as a service model

In our system, a two-level auditing model is adopted: each user records his operations in a user operation table (UOT), which is referred to as a local trace of operations in this paper. Local auditing can be performed independently by each user with his own UOT; periodically, an auditor is elected from the audit cloud. In this case, all other users will send their UOTs to the auditor, which will perform global auditing with a global trace of operations. We simply let each user become an auditor in turn, and we will provide a more comprehensive solution. The dotted line in the audit cloud means that users are loosely connected. That is, users will communicate to exchange messages after executing a set of reads or writes, rather than communicating immediately after executing every operation. Once two users finish communicating, a causal relationship on their operations is established.

B. User Operation Table:

User operation table is maintained by user for recording their local operations. Each record in the UOT contains mainly three elements: operation, physical vector and logical vector. While issuing any operation a user will record his operation

as well as physical and logical vector in his UOT. The physical clock in the physical vector continuously increased by the client and logical clock in the logical vector increased by the client only when an event happens. By using UOT, take trace of operations for verifying global consistency.

C. Two Level Auditing Structure:

In CaaS model a two level auditing structure only requires a loosely synchronized clock for ordering their operations. This two level auditing structure is used to verify the consistency properties. In the first level each user perform local auditing separately using their own UOT. In this level local consistency is performed. There are two types of local consistency. They are monotonic read consistency and read-your-write consistency.

Monotonic read consistency: If any process read the value of data X as well as successive read on data X then the user must read the same value or more recent value

Read-your-write consistency: If write of process on data X will be seen by successive reads on data X then the user always reads the same process

A local consistency algorithm is used to verify the local consistency properties. It is an online algorithm.

In the second level, causal consistency is verified. In this level an auditor will be selected from the audit clouds which can able to perform global consistency using global trace of operations.

Causal consistency: all causally related read/write operations were executed in an order that reflects their causality, all concurrent operations may be seen in different orders.

A global consistency algorithm is used to verify causal consistency. It is an offline algorithm. Global consistency strategy is performed by constructing a directed graph. To observe whether this graph is a DAG or not. If it is a DAG then we can conclude it preserve causal consistency.

D. Gossip Based System:

If cycle is present in the constructed graph , we can able to understand that is not a DAG. From this graph a user can conclude that there have an inconsistency in the cloud, the data cloud does not provide promised level of consistency. To avoid inconsistency in the cloud, the optimal solution is NP hard problem.

In NP hard problem, each and every node in the network communicate with each other and send their data or status. Suppose there are 10 nodes, each node send their data or status to other 9 nodes. So complexity become 10^{10}, it is not feasible. So we introduce an another technique called gossip based update. Gossip is a style of computer to computer communication protocol inspired by the form of gossip seen in social network. Modern distributed system often use gossip protocol to solve problems that might be difficult to solve in other ways. The concept of the gossip communication can be illustrated by the analogy of office workers spreading rumours.

a. Gossip Protocol:

Gossip is a peer to peer communication protocol in which nodes periodically exchange state information about themselves and about other nodes the know about. The gossip process runs every second and exchange state messages with up to three other nodes in the cluster. The nodes exchange information about themselves and about the other nodes that they have gossiped about. So all nodes quickly learn about all other nodes in the cluster. A gossip messages has the version associated with it, so that during a gossip exchange ,older information is over written with the most current state for a particular node.

Let's take an example, let's assume there are five machines A, B, C, D, & E in a network, all connected to each other.

1. B contacted A and shared its state/data. Now A has= A State/Data+ B State/Data.

2. E also contacted A and shared its state/data. Now A has= A State/Data+ B State/Data + E State/Data

3. C contacted D and shared its state/data. Now D has =D State/Data+ C State/Data.

Now

4. When D contacted A and share its state/Data, A will have=A State/Data+ B State/Data + E State/Data + D State/Data+ C State/Data

Similarly when A shares its data with D, D will have=A State/Data+ B State/Data + E State/Data + D State/Data+ C State/Data

ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 4, Issue 3, pp: (153-158), Month: July - September 2016, Available at: www.researchpublish.com

5. Now next time whenever A contacts B, E both B and E will have =A State/Data+ B State/Data + E State/Data + D State/Data+ C State/Data

same is the case with C.

6. While getting data each node checks the time stamp and accepts the latest data and rejects old ones.

As shown in the previous example the machines share each other's state using gossip and in no time all the machines will be aware of the state of all other machines. So if any machine goes down, others in the network will come to know soon. They will keep pinging the down machine as will in future, and hence whenever it comes up, its live status will be updated in all other machines quickly.

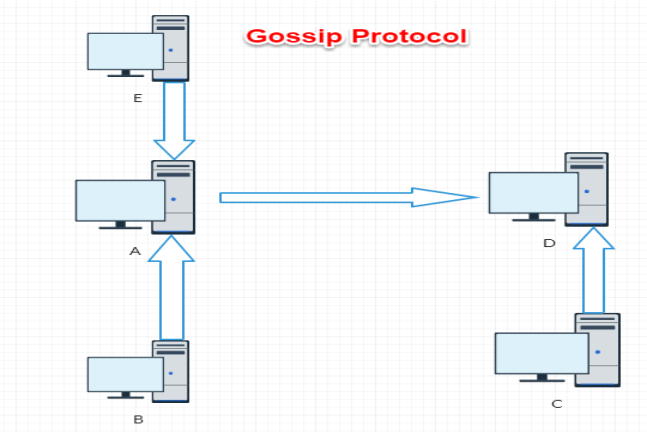

Fig.4: Gossip protocol

The operation of a gossip protocol is as follows.

- User 1 initializes gossiping by sending a request message to user 2
- After the reception of a request by user 2, it sends a response back to user 1
- User1, based on the response, may either synchronize its clock to the clock of user2, send a feedback message to user2, or ignore the message.
- If user2 receives the feedback message, it may use it to synchronize its clock to the clock of user1.

V. PERFORMANCE ANALYSIS

In this section compare the distributed system without gossiping time protocol (old) and the distributed system with gossiping time protocol (new).

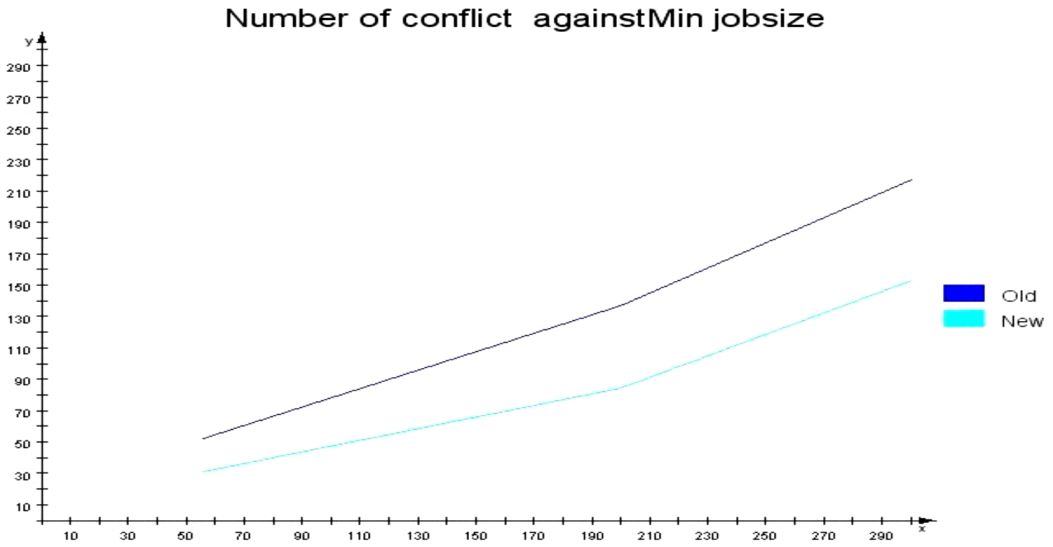

Fig.5: Comparison of cloud storage system without gossiping and with gossiping

VI. CONCLUSION

In this paper, presented a consistency as a service (CaaS) model and a two level auditing structure to help users verify whether the data cloud maintained by the cloud service provider (CSP) is providing the promised level of consistency or not. Two consistency auditing algorithms are presented to check different types of consistency properties. A gossip based system is used to avoid the consistency to a maximum level. Through the CaaS model with gossiping time protocol the users can determine the quality of cloud services and select a right CSP from various CSPs, and also avoid the inconsistencies (not completely) with gossip based system.

VII. ACKNOWLEDGEMENT

I would like to extend my thankfulness to the reference authors, as well as evaluator of my paper.

REFERENCES

[1] M. Jelasity, A. Montresor, and O. Babaoglu, "Gossip-based aggregationin large dynamic networks," ACM Trans. Computer Syst., vol. 23, no. 3, pp. 219–252, 2005.

[2] F. Wuhib, R. Stadler, and M. Spreitzer, "Gossip-based resource management for cloud environments," in 2010 International Conference on Network and Service Management.

[3] F. Wuhib, M. Dam, and R. Stadler, "A gossiping protocol for detecting global threshold crossings," IEEE Trans. Network and Service Management, vol. 7, no. 1, pp. 42–57, Mar. 2010.

[4] R. Yanggratoke, F. Wuhib, and R. Stadler, "Gossip-based resource allocation for green computing in large clouds," in 2011 International Conference on Network and Service Management.

[5] G. Pacifici, W. Segmuller, M. Spreitzer, and A. Tantawi, "Dynamic estimation of CPU demand of web traffic," in valuetools '06: Proceedings of the 1st international conference on Performance evaluation methodolgies and tools. New York, NY, USA: ACM, 2006, p. 26.

[6] D. Carrera, M. Steinder, I. Whalley, J. Torres, and E. Ayguade, "Utility-based placement of dynamic web applications with fairness goals," in Network Operations and Management Symposium, 2008. NOMS 2008. IEEE, april 2008, pp. 9 –16.

[7] S. Voulgaris, D. Gavidia, and M. van Steen, "CYCLON: Inexpensive membership management for unstructured p2p overlays," Journal of Network and Systems Management, vol. 13, no. 2, pp. 197–217, 2005.

[8] C. Tang, M. Steinder, M. Spreitzer, and G. Pacifici, "A scalable application placement controller for enterprise data centers," in WWW '07: Proceedings of the 16th international conference on World Wide Web. New York, NY, USA: ACM, 2007, pp. 331–340.

[9] H. Kellerer, U. Pferschy, and D. Pisinger, Knapsack problems. Springer-Verlag, 2004.

[10] G. B. Dantzig, "Discrete-Variable Extremum Problems," OPERATIONS RESEARCH, vol. 5, no. 2, pp. 266–288, 1957.