HEURISTIC AUDITING STRATEGY FOR CLOUD CONSISTENCY

Dr. Siddaraju
Professor and Head ,Department Of CSE,
Dr. Ambedkar Institute of Technology, Bangalore

Yogesh K.
Student of M.Tech CSE
Dr. Ambedkar Institute of Technology, Bangalore

Abstract :*In this work we provide Data Integrity verification and consistency services for different data storage systems in cloud by supporting public auditability and data dynamics. These two components are integrated to solve internal and external threats for data security. Consistency is the property that guarantees that all parties interested in a given piece of information get updated each time it changes. In the context of massive distributed systems that now operate on a world scale, and looking into cloud computing in particular, consistency needs and integrity of data to be traded for decision support activity and get efficient item set result based on the caas. In this paper we discuss on whether cloud provides promised level of consistency or not, local and global auditing mechanisms are used for ordering operations and heuristic auditing strategy (HAS) to reveal as many violations possible and quantify them and enable to maintain data integrity proof by read and write permission that are provided by data owner to the data consumer (end user) for accessing the cloud storage system. The data owner can also audit the data integrity in the corresponding cloud for verifying whether the data is safe or not.*

Index Terms*- Cloud storage, Data Owner, Cloud server, Audit cloud, Data consumer, consistency as a service(caas), Data integrity,and heuristic auditing strategy(HAS).*

I. INTRODUCTION

As cloud computing applications are rowing and becoming massively distributed, data is stored on many servers.Achieving high availability and scalable performance requires some sort of data replication technique. However data replication is not without challenges that need to be addressed, especially with regards to consistency. Data needs to be updated in several locations and a problem thus arises when one or more of these locations are temporarily not accessible. Section 2 provides various definitions of consistency and section 3 discusses the concept of cloud computing. Section 4 goes into detail on consistency trade-offs and describes different consistency models and associated levels of inconsistency.

A. CONSISTENCY – DEFINITION

From a database point of view, "Consistency states that only valid data will be written to the database. If, for some reason a transaction is executed that violates the database consistency rules, the entire transaction will be rolled back and the database will be restored to a state consistent with those rules. On the other hand, if a transaction successfully executes, it will take the database from one state that is consistent with the rules to another state that is also consistent with the rules" [11].Basically, this means that the output of a transaction is committed when the transaction abdicates the right to undo the changes made resulting in that output thereby making the new value available to all transactions. However, the context of this paper is massively distributed systems where data is generally replicated to achieve high availability and improve performance. Here, the collection of replicas is said to be consistent if all the replicas are the same. This means that if a read operation is carried out at any of the replicas or copies; it will return the same result. Also if a replica is modified or updated, all other replicas will be updated as well no matter which replica the operation originated from. This type of consistency is most times referred to as strong or strict consistency. Other types are discussed in subsequent sections.

B.CLOUD COMPUTING - AN OVERVIEW

"Cloud Computing is a style of computing in which IT related capabilities are provided "as a service", allowingusers to access technology enabled services from the internet(referred to as the cloud) without knowledge of,expertise with, or control over the technology infrastructure that supports them." The focus is on sharing data andcomputations over a scalable network of nodes which are end users, data centers and web services. The mainidea is to use the existing infrastructure in order to bring all feasible

services to the cloud and make it possible toaccess those services regardless of time and location.Based on the functionality offered, there are three main types of cloud computing services[x].

i. SOFTWARE-AS-A-SERVICE (SAAS) - This offers already developed and successfully launched applications.Hence users just need a computer or server to download the application and access to the internet to runthe application instead of purchasing the necessary hardware or software. Examples include GoogleCalendar.

ii. PLATFORM-AS-A-SERVICE (PAAS)- This offers facilities required for the complete development and delivery ofweb applications and services through the internet. So it basically offers an API which can be used by theapplication developer. Examples include Google App Engine

iii. INFRASTRUCTURE-AS-A-SERVICE (IAAS) -This provides an environment for the delivery of infrastructure. Hence acustomer purchases required resources through an outsourced service rather than purchasing servers,hardware, storage and networking components. Examples include Amazon EC2Generally, the main drivers of cloud computing are economies of scale, reduction of total IT spend withoutcompromising quality and gaining flexibility and speed in implementation.

II. CONSISTENCY TRADE-OFFS AND ITS LEVELS

As cloud computing applications are growing and becoming massively distributed, the number of servers storing the same data is growing. This replication of data is to enhance availability and reliability so that data can still beaccessed even when one system or more is unavailable and to improve performance as multiple copies of datahelp scale a system to larger numbers of clients and geographic areas. There are two basic types of consistency models[x] namely data centric and client centric. A consistency model isbasically a set of rules to be obeyed by processes while accessing data.

a) DATA CENTRIC MODEL:This type of model concentrates on consistency from a system wide perspective of the storage system. It assumes that concurrent processes may be updating the storage system. This model can be further broken into two types - models that do not use synchronization operations and those that do. Examples include strict consistency, causal consistency, and sequential consistency.

b) CLIENT CENTRIC:This type of model concentrates on consistency from a single client perspective with respect to the data stored by that client. It assumes that concurrent updates can be easily resolved or are not being made at all. Examples include monotonic read, monotonic write, read your writes and writes follow read. Also consistency can be seen from two perspectives namely: client/developer view which has to do with how they observe updates and from the server view which involves the flow of updates through the system and the consistency guarantees with respect to these updates.

1) CLIENT VIEW

i. Strict Consistency- This is the strictest possible model and it does not use synchronization operations. It guarantees that if any data is being accessed, the value returned must correspond to the result of the most recent update to that data. However, this is impossible to implement in a distributed system because theabsolute time ordering of all shared access matters.

ii. Weak Consistency- This model uses synchronization operations to synchronize all local copies of the storage system. This involves using synchronization variables (which can be seen by all processes in the same order) to propagate writes to and from a machine at appropriate points. It does not guarantee thatsubsequent accesses will return the updated value as this depends on a number of conditions that need to be met. Basically, access to the synchronization variable is not allowed until all pending write operations are completed and no new read/write operation is allowed if there is an ongoing synchronization operation.

iii. Eventual Consistency - This is a specific form of weak consistency. It guarantees that if no new updates are made to a particular data, eventually all accesses to that data will return the last updated value. Eventual consistency works just fine for replicated data if clients always access the same replicas but poses a problem when different replicas are involved. This problem can however be lightened by introducing client centric consistency models. The two most desirable are monotonic reads and read your writes. Monotonic reads guarantee that if a process reads the value of a data item, subsequent accesses by that process will return the same value or a more recent value while read your writes guarantee that if a process updates a data item, subsequent accesses by the same process will always return the updated value.

2) SERVER VIEW

First a few definitions,

N = the number of nodes that store replicas of the data.

W = the number of replicas that need to acknowledge the receipt of the update before the updatecompletes.

R = the number of replicas that are contacted when a data object is accessed through a read operation

Strong consistency can be guaranteed if W+R>N as this means there is always an overlap between the write andread sets. On the other hand Weak or Eventual consistency occurs if W+R <= N. Here the system is vulnerable toreading from nodes that have not yet received updates. These configurations however depend on the focus of thesystem. If the focus is fault tolerance, the configuration is usually set to N=3, W=2, and R=2. In the case ofconsistency, it is set to W=N for updates.

III. HEURISTIC AUDITING STRATEGY FOR CLOUD CONSISTENCY

From the auditing process in the CaaS model, we observe that only reads can reveal violations by their values. Therefore, the basic idea of our heuristic auditing strategy (HAS) is to add appropriate reads for revealing as many violations as possible. We call these additional reads auditing reads. As shown in Fig.1. HAS divides physical time into L timeslices, where l timeslices constitute an interval. Each timeslice is associated with a state, which can be marked witheither normal or abnormal. A normal state means that there is no consistency violation, and an abnormal state means that there is one violation in this timeslice.

HAS determines the number of auditing reads in the (i+1)-th interval, based on the number of abnormal states in the i-th interval. Let ni denote the number of auditing reads in interval i. HAS determines ni+1, which is the number of auditing reads in the next interval with Eq. 1_

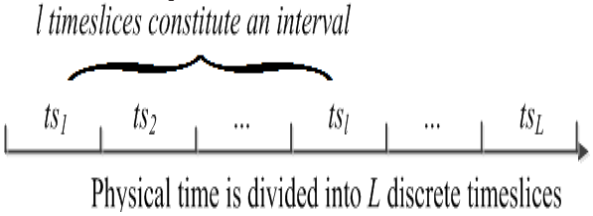


Fig 3.1: time slice

$n_{i+1} = \min(l, k \times n_i), n_i \geq \alpha$ (1)

$n_{i+1} = \max(1, 1k \times n_i), n_i < \alpha$ (2)

where k is a parameter that is used to adjust the value of ni+1, l is the number of time slices in an interval, and α is a threshold value that is used to determine whether the number of auditing reads in the next round should be increased by k times or be reduced to 1/k, compared to the number of auditing reads in the current round. Specifically, given a threshold value α, if a user issues ni auditing reads and reveals more than α violations in interval i, in interval i + 1, the user will issue ni+1 = min(l, k ∗ni) auditing reads; that is, each timeslice will be issued, at most, one auditing read, and the maximal number of auditing reads will not exceed l. Otherwise, the user will issue ni+1 =max(1, 1×ni) auditing reads, that is, each interval will beissued at least one auditing read. Since the number of auditingreads should be an integer, 1k×ni is actually the abbreviation of (1/k ×ni). Suppose that the SLA stipulates that the audit cloud can gain s (e.g., monetary compensation) from the data cloud if a consistency violation is detected, and that the audit cloud will be charged r for a read operation. If after executing n auditing reads, the users reveal v violations, then the earned profits P can be calculated by P = s ∗ v − r ∗ n. Under the CaaS model, consistency becomes a part of the SLA, the users can obtain proportional compensation from the CSP, by revealing consistency violations and quantifying the severity of the violations. We believe that the CaaS model will help both the CSP and the users adopt consistency as an important aspect of cloud services offerings.design to seamlessly integrate these two important components for data storage service remains an open challenging task in Cloud Computing.

IV. 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

DISADVANTAGES OF EXISTING SYSTEM:

1 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.
2 Second, there do exist various motivations for CSP to behave unfaithfully toward the cloud users regarding their outsourced data status.
3 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.
4 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 not accessed data and might be too late to recover the data loss or damage.
5 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.

V. 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 a real cloud deployment to validate HAS.

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

VI. System Architecture

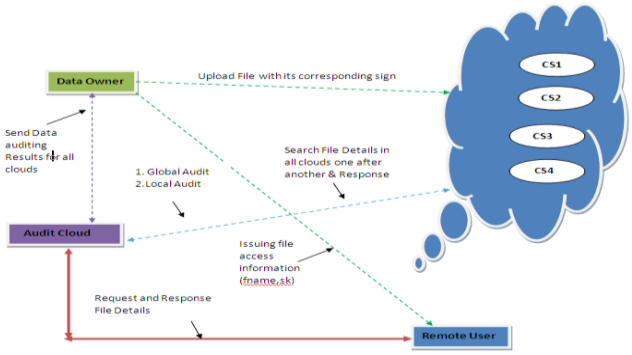


Fig 4.1: system architecture

- Data Owner

In this module, the data owner uploads their data in the cloud server. For the security purpose the data owner encrypts the data file and then store in the cloud. The Data owner can have capable of manipulating the encrypted data file. The data owner will send Meta data to Audit cloud. In audit cloud raw or metadata information is available for auditing and data integrity checking purpose. Data owner will create an end user and the data owner can set the access permission (read or write) to user.

b) Data Auditing and Verification

The data owner can also audit the data integrity in the corresponding cloud for verifying whether the data is safe or not. If the data is not safe then he will delete the data and re upload the data to the corresponding cloud server.

- Cloud Server s

The cloud server is responsible for data storage and file authorization for an end user. The data file will be stored with their tags such as file name, secret key, digital sign, and owner name. The data file will be sending based on the privileges. If the privilege is correct then the data will be sent to the corresponding user and also will check the file name, end user name and secret key. If all are true then it will send to the corresponding user or he will be captured as attacker. The cloud server can also act as attacker to modify the data which will be auditing by the audit cloud.

Audit cloud
Audit cloud is a cloud which is responsible for handling local and global auditing. In local auditing this system will capture the file attackers from all the clouds such as cs1, cs2, cs3 and also block the corresponding file hacker. It will also audit the access privileges such as read or write operations. In

the global auditing, this system will audit the data integrity and the proof will be given to the corresponding end user. Also, it will capture the entire cloud file accessing while tracing from one after another cloud from the corresponding end user.

- Data Consumer(End User)

  The data consumer is nothing but the end user who will request and gets file contents response from the corresponding cloud servers. If the file name and secret key, access permission is correct then the end is getting the file response from the cloud or else he will be considered as an attacker and also he will be blocked in corresponding cloud. If he wants to access the file after blocking he wants to UN block from the cloud.

- Attacker
  Attacker is one who is integrating the cloud file by adding malicious data to the corresponding cloud. They may be within a cloud or from outside the cloud. If attacker is from inside the cloud then those attackers are called as internal attackers. If the attacker is from outside the cloud then those attackers are called as external attackers.

c) Data Flow diagram

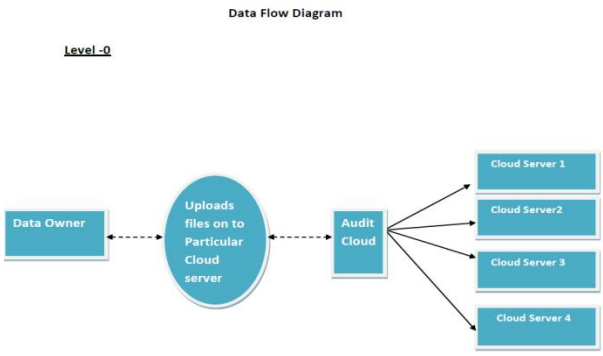


Fig 4.2 level 0 data flow diagram

Data Owner uploads files on to particular cloud server and sends metadata information to the audit cloud.

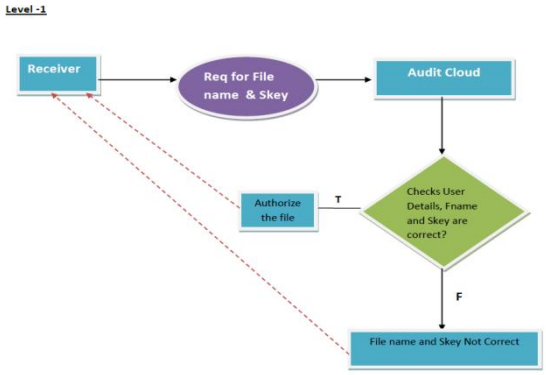


Fig 4.3: level 1 of data flow diagram

Level 1: Receiver request for the required file by using file name and secret key to the audit cloud. The audit cloud would verify the user details such as file name and secret key. If the given filename and secret key is correct it would allow the user to access & authorize the file .If the file name and secret key is not correct then the user cannot access the file.

Level

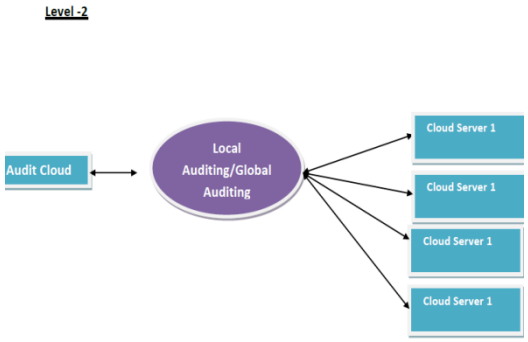


Fig 4.4: level 2 of data flow diagram

Here audit cloud would perform local auditing and global auditing. In local auditing this system will capture the file attackers from all the clouds such as cs1, cs2, cs3 and also block the corresponding file hacker. It will also audit the access privileges such as read or write operations. In the global auditing, this system will audit the data integrity and the proof will be given to the corresponding end user. Also, it will capture the entire cloud file accessing while tracing from one after another cloud from the corresponding end user.

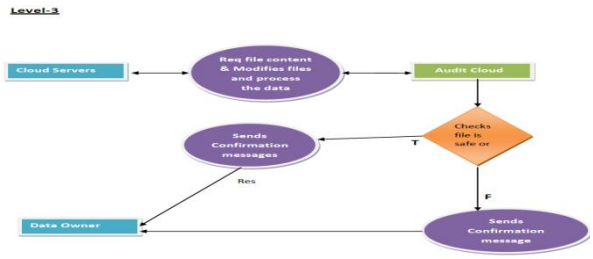


Fig 4.5: level 3 of data flow diagram

Level 3: Here the cloud server would request file and modifies the file contents and sends process data to the audit cloud. The audit cloud would check file is safe or not and sends confirmation messages to the data owner about the modified files.and filename. The cloud server would store files,helps to view files, modify file contents.

VII. Verifying the Cloud Consistency

In this section, we see the algorithms used for maintaining consistency of cloud. Advanced Encryption Standard(AES)

It is a symmetric block encryption algorithm with a block length of 128 bits and support for key lengths of 128,192, and 256 bits. Evaluation criteria include security, computational efficiency, memory requirements, hardware, software suitability, and flexibility.

The Secure Hash Algorithm(SHA) developed by the National Institute of Standards and Technology(NIST) . This algorithm takes as input message with maximum length of less than 2^128 bits and produces as output a 512-bit message digest. It is used for secure exchange of data and messages.

Local consistencyauditing is an online algorithm(Alg1) where each user will record all his operations in his UOT. While issuing a read operation, the user will perform local auditing independently.

Global consistency auditing is an offline algorithm (Alg2). 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.

User Operation Table(UOT) 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 operation op is either a write W(K, a) or a read R(K, a), where W(K,a) means writing the value a to datathat is identified by key K, and R(K, a) means reading data that is identified by key K and whose value is a. As we call W(K, a) as R(K, a)'s dictating write, and R(K, a) as W(K, a)'s dictated read. We assume that the value of each write is unique. This is achieved by letting a user attach his ID, and current vectors to the value of write. Therefore, we have the following properties: (1) A read must have a unique dictating write. A write may have zero or more dictated reads. (2) From the value of a read, we can know the logical and physical vectors of its dictating write. Each user will maintain a logical vector and a physical vector to track the logical and physical time when an operation happens, respectively. Suppose that there are N users in the audit cloud. A logical/physical vector is a vector of N logical/physical clocks, one clock per user, sorted in ascending order of user ID. For a user with IDi where $1 \leq i \leq N$, his logical vector is <LC,LC2, . . . , LCN >, where LCiis his logical clock, and LCj is the latest logical clockof user j to his best knowledge; his physical vector is< PC1, PC2, . . . , PCN >, where PCi is his physical clock,and PCj is the latest physical clock of user j, to the best ofhis knowledge. 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. 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.

Algorithms used are:

1. AES - For Data Encryption and Decryption.
2. Sha1 - Secure Hash Algorithm - for digital signature
3. Local consistency auditing - For auditing local consistency - attackers while accessing file and checking read or write

Algorithm 1:Local consistency auditing

S1: Initial UOT with ∅

while issue an operation op do

S2: if op = W(a) then

record W(a) in UOT

S3: if op = r(a) then

W(b) ∈ UOT is the last write:
elseif W(a) → W(b) then
Read-your-write consistency is violated
R(c) ∈ UOT is the last read
S4: if W(a) → W(c) then
Monotonic-read consistency is violated
record r(a) in UOT

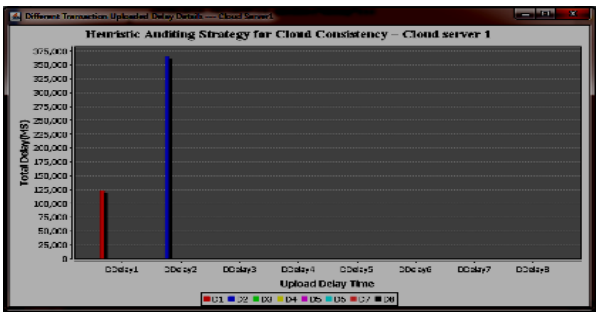
S1- Check the user request.
S2 - Check the consistency type(Read or Write).
S3 -Allow the file accessing based on read or write users.
S4- Capture the attackers if the rule is violated.
4.Global Consistency - Data integrity and auditing and checking all the clouds for accessing file

Algorithm 2:Global consistency auditing
S1: Each operation in the global trace is denoted by a vertex
S2: 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
S3: 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
S4: Check whether the graph is a DAG by topological sorting

S1- Check the number of users file request
S2 - Check file in all cloud(here cloud called as edges) and register all cloud tracing in audit cloud
S3 -Allow the file if file is not integrated
S4- Capture the file auditing if the file is modified

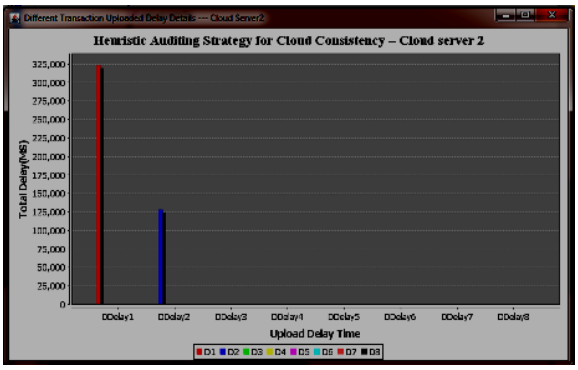
VIII. EVALUATION

Here we would verify the effectiveness of HAS, we conduct experiments on synthetic and real violation traces.



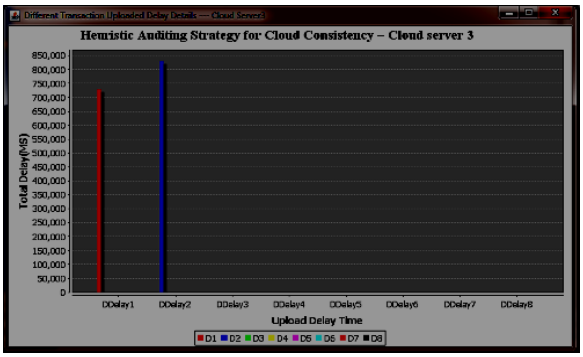
Graph 1: total delay

Graph 1. Here the graph is plotted for total delay (ms) versus total upload time for cloud server1 where the upload delay time is divided into 8 units .The cloud server1 takes a time of 125000ms to upload a file1 onto cloud with upload delay of DDelay1 and it takes a time of 350000ms to upload file2 onto the cloud with upload delay of DDelay2.



Graph 2: total delay versus upload time for server 2

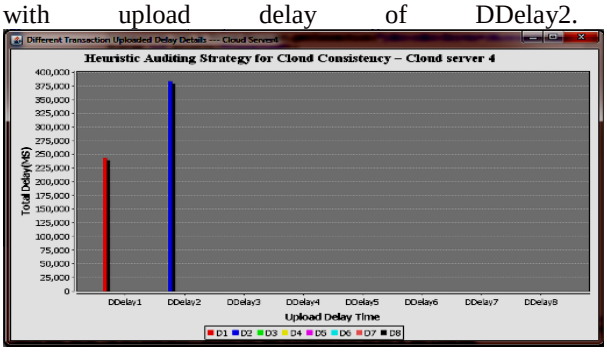
Graph 2. Here the graph is plotted for total delay (ms) versus total upload time for cloud server2 where the upload delay time is divided into 8 units .The cloud server2 takes a time of 325000ms to upload a file1 onto cloud with upload delay of DDelay1 and it takes a time of 175000ms to upload file2 onto the cloud with upload delay of DDelay2.



Graph 3: time delay versus upload time for server 3

Graph 3. Here the graph is plotted for total delay (ms) versus total upload time for cloud server3 where the upload delay time is divided into 8 units .The cloud server3 takes a time of 725000ms to upload a file1 onto cloud with upload delay of DDelay1 and it takes a time of 825000ms to upload file2 onto the cloud

with upload delay of DDelay2.



Graph 4: time delay versus upload time for server 4

Graph 4. Here the graph is plotted for total delay (ms) versus total upload time for cloud server4 where the upload delay time is divided into 8 units .The cloud server3 takes a time of 275000ms to upload a file1 onto cloud with upload delay of DDelay1 and it takes a time of 385000ms to upload file2 onto the cloud with upload delay of DDelay2

The above graphs we can observe the delays to upload the files to various servers and their level of maintaining consistencies for different file size using heuristic auditing strategy. Itis also observed that HAS is impacted by the length of the file, threshold value i.e delay .HAS can reveal 90% of violations when ddelay1 and reveal 81% when ddelay2. HAS can detect almost all of violations when threshold and length are chosen properly.

IX. CONCLUSION

In this paper, we present enabling data integrity proof and consistency services over multi cloud system using Heuristic auditing strategy which helps in revealing violations as much as possible. The cloud consistency model and local auditing, global auditing that helps users to verify the cloud service provider (CSP) provides the promised consistency or not and quantify the severity of the violations. This helps the users to choose the required cloud as per their requirements based on service level agreement(SLA). Here auditing is performed to know whether the files are safe or not, if any such violations are found it would be revealed. The main purpose is to provide security for the data that is stored on the cloud by the users through various encryption and decryption methods. In our future work, we can focus on generating logs on data integrity via cloud.

X. 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.

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

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

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

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

[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.

