IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 19, Issue 2, Ver. IV (Mar.-Apr. 2017), PP 73-77
www.iosrjournals.org

Auditing Cloud Consistency

Presented By: Prajakta Suryawanshi
ME Research Scholar
Guided By: Prof.Pranay Saraf
Industry Mentor: Harender Singh

Abstract: *Given model consist of large data cloud and small audit clouds.The data cloud is maintained by a CSP, and an audit cloud consists of a group of users that cooperate on a job, e.g., a document or a project. A service level agreement (SLA) will be engaged between the data cloud and the audit cloud, which will stipulate what level of consistency the data cloud should provide, and how much (monetary or otherwise) will be charged if the data cloud violates the SLA. The implementation of the data cloud is opaque to all users due to the virtualization technique. Thus, it is hard for the users to verify whether each replica in the data cloud is the latest one or not. We allow the users in the audit cloud to verify cloud consistency by analyzing a trace of interactive operations. Unlike their work, we do not require a global clock among all users for total ordering of operations. A loosely synchronized clock is suitable for our solution.*

Index: *Consistency as service, two level auditing, Cloud Storage, Heuristic Auditing strategy.*

I. Introduction

In this project is to upload and download a file from cloud. In this project, a model that we designing a novel consistency as a service (Camas) model, which consists of a large data cloud and multiple small audit clouds. Also two-level auditing architecture, which only requires a loosely synchronized clock in the audit cloud. By designing algorithms to quantify the severity of violations with two metrics: the commonality of violations, and the staleness of the value of read. We devise a heuristic auditing strategy (HAS) to reveal as many violations as possible.

While providing cloud consistency, the following objectives are to be met:

- Understanding the novel consistency as a service (Camas) model provided by the cloud service provider.
- Checking weather cloud provides promised consistency or not. The cloud computing solution should provide basic consistency as service.
- Maintain synchronized clock at audit clouds that responsible for
- Service Availability.

  The cloud storage is services have become in commercially popular due to their overwhelming advantages in a cloud computing work .A key problem of using the replication of multiple file transmit ion on the networks.

Technique in clouds is that it is very expensive to achieve strong consistency on a worldwide scale. In this project we first present a novel consistency as a service (Camas) model in the network, which consists of a large data cloud and multiple small audit clouds for transition of files. In those (Camas) model, a data cloud is maintained by cloud service provider, and group of users that constitute an audit cloud can verify whether the data cloud provides the promised level of consistency or not in a transition medium. We propose a two-level auditing architecture in it, which only requires a loosely synchronized clock in the audit cloud of a network. Then, we design algorithms to quantify the severity of violations with two metrics: first one is the commonality of violations, and second one is the staleness of the value of a read. Finally, we devise a heuristic auditing strategy (HAS) to reveal as many violations as possible in it. Extensive experiments were performed using a combination of simulations and real cloud deployments to validate have.

II. Related Work

A cloud is essentially a large-scale distributed system where each piece of data is replicated on multiple geographically distributed servers to achieve high availability and high performance. Thus, we first review the consistency models in distributed systems Data-centric consistency model considers the internal state of a storage system, i.e., how updates flow through the system and what guarantees the system can provide with respect to updates. However, to a customer, it really does not matter whether or not a storage system internally contains any stale copies. As long as no stale data is observed from the client's point of view, the customer is satisfied. Therefore, client-centric consistency model concentrates on what specific customers want, i.e., how the customers observe data updates. Finally, we review the work on verifying the levels of consistency provided by the CSPs from the users' point of view. Existing solutions can be classified into trace-based verifications and benchmark-based verifications. A register is safe if a read that is not concurrent with any write returns the value of the most recent write, and a read that is concurrent with a write can return any value. A register is regular if a read that is not concurrent with any write returns the value of the most recent write, and a read that is concurrent with a write returns either the value of the most recent write, or the value of the concurrent write. However, we focus on different consistency semantics in commercial cloud systems, where a loosely synchronized clock is suitable for our solution. Benchmark-based verifications focus on benchmarking staleness in a storage system. The results of justify our two-level auditing structure. Presents a client-centric benchmarking methodology for understanding eventual consistency in distributed key value storage systems.

III. Our Approach

Main Objective of this project is to upload and download a file from cloud. While downloading file from cloud, we are checking whether the file is updated version or not. Also finding out time require updating the file within multiple servers.

- Providing the upload and download tab with brows feature.
- Button to submit the file on cloud.
- Text Label to display warnings, exceptions and errors.
- Message boxes to display messages to user such as operation successful or not.

Assumptions:

In general it has been assumed that the user has complete knowledge of the system that means user is not a naïve user.

The software is as user friendly as possible but at the same time keeping in minds user requirements. All the PCs must have Windows OS with .NET framework on it and configured to form a cluster.

Dependencies:

It depends that the one should follow the international standards for the generating the User ID & should fill the related information in the proper format.

Responses for Administrator: The administrator can Login and Logout. When administrator logs into the Software, the system will check for validity of login .If the Login and passwords are valid, the response to this action is the administrator will be able to view, add, deleting and all other functions that can be performed on the database.

IV. Consistency as Services:

Eventual consistency is a consistency model used in distributed computing to achieve high viability that informally guarantees that ,if no new updates are made to a given data item eventually all accesses to the item will return the last updates value.

A system that has achieves eventual consistency is often said to have converged or replica convergence.

V. Two-Level Auditing Structure:

Cloud computing security or, more simply, cloud security refers to a broad set of policies, technologies, and controls deployed to protect data, applications, and the associated infrastructure of cloud computing. Cloud computing and storage provides users with capabilities to store and process their data in third-party. Organizations use the cloud in a variety of different service models and deployment models. Security concerns associated with cloud computing fall into two broad categories: security issues faced by cloud providers and security issues faced by their customers the responsibility is shared, however. The provider must ensure that their infrastructure is secure and that their clients' data and applications are protected, while the user must take measures to fortify their application and use strong passwords and authentication measures.

Cloud security architecture is effective only if the correct defensive implementations are in place. Efficient cloud security architecture should recognize the issues that will arise with security management.

Deterrent controls:

These controls are intended to reduce attacks on a cloud system. Much like a warning sign on a fence or a property, deterrent controls typically reduce the threat level by informing potential attackers that there will be adverse consequences for them if they proceed.

Preventive controls:

Preventive controls strengthen the system against incidents, generally by reducing if not actually eliminating vulnerabilities. Strong authentication of cloud users, for instance, makes it less likely that unauthorized users can access cloud systems, and more likely that cloud users are positively identified.

Detective controls:
Detective controls are intended to detect and react appropriately.

Preventive controls:
Preventive controls strengthen the system against incidents, generally by reducing if not actually eliminating vulnerabilities. Strong authentication of cloud users, for instance, makes it less likely that unauthorized users can access cloud systems, and more likely that cloud users are positively identified.

Detective controls:
Detective controls are intended to detect and react appropriately to any incidents that occur. In the event of an attack, a detective control will signal the preventative or corrective controls to address the issue. System and network security monitoring, including intrusion detection and prevention arrangements, are typically employed to detect attacks on cloud systems and the supporting communications infrastructure.

Corrective controls:
Corrective controls reduce the consequences of an incident, normally by limiting the damage. They come into effect during or after an incident. Restoring system backups in order to rebuild a compromised system is an example of a corrective control.

VI. Heuristic Serach Algorithm:

Heuristic is a technique designed for solving a problem more quickly when classic methods are too slow, or for finding an approximate solution when classic methods fail to find any exact solution. A heuristic function, also called simply a heuristic, is a function that ranks alternatives in search algorithms at each branching step based on available information to decide which branch to follow. For example, it may approximate the exact solution. The objective of a heuristic is to produce a solution in a reasonable time frame that is good enough for solving the problem at hand. This solution may not be the best of all the actual solutions to this problem, or it may simply approximate the exact solution. But it is still valuable because finding it does not require a prohibitively long time.

Heuristics may produce results by themselves, or they may be used in conjunction with optimization algorithms to improve their efficiency.

Results in theoretical computer science make heuristics the only viable option for a variety of complex optimization problems that need to be routinely solved in real-world applications.

Heuristics underlie the whole field of Artificial Intelligence and the computer simulation of thinking, as they may be used in situations where there is no known algorithm.

The trade-off criteria for deciding whether to use a heuristic for solving a given problem include the following:

- Optimality: When several solutions exist for a given problem, does the heuristic guarantee that the best solution will be found? Is it actually necessary to find the best solution?
- Completeness*:* When several solutions exist for a given problem, can the heuristic find them all? Do we actually need all solutions? Many heuristics are only meant to find one solution.
- Accuracy and precision: Can the heuristic provide a confidence interval for the purported solution? Is the error bar on the solution unreasonably large?
- Execution time: Is this the best known heuristic for solving this type of problem? Some heuristics converge faster than others. Some heuristics are only marginally quicker than classic methods.

VII. Modules Details

1) LOGIN PHASE:
The first module is the login phase,in this phase firstly we define the user details in this we define the “username” and “password”. Than enter both of the details and click the login option and if the user is new than click the register option.

2) REGISTER PHASE:
In this phase we declare the name, mobile number, email id, username, and password, confirm password and click the register and go to the next step. If the details is wrong the click the option back for re-registration.

3) START THE SERVER
In the we click two option first is my files and second is upload option. The we upload the files on server and then we check the updating replicas is found or not with the use of heuristic algorithm.

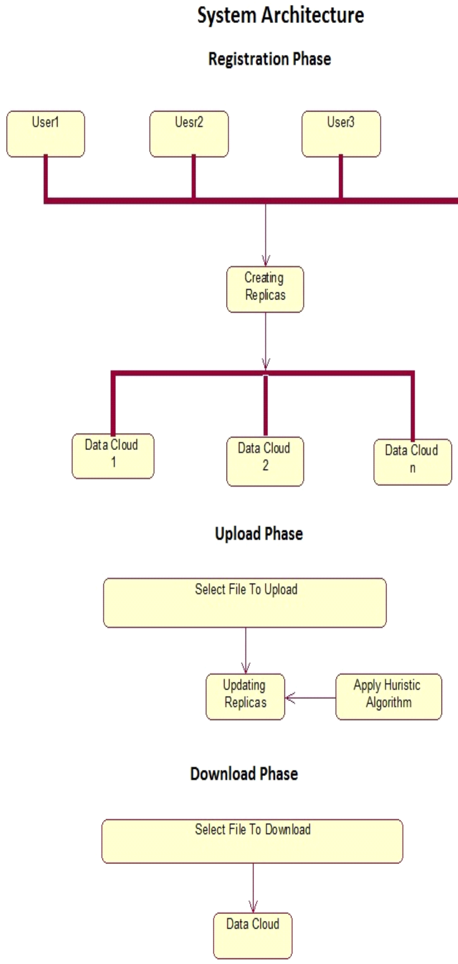

Fig. System Architecture

VIII. Collusion

We presenting research results is a vital aspect of research work in based on cloud computing. The communication of research findings provides a valuable opportunity to inform others of a current investigation in a cloud and it and can lead to future speaking opportunities at conferences properly in this, grants for future research projects, school and business meetings and offer natural connections to new job opportunities in cloud by using cloud computing. Firstly my discussion will highlight major elements in preparing research presentations that will help other researchers to best represent their research while effectively meeting audience expectations. An emphasis will be placed on action research projects that are growing more popular in today's graduate education programs easily. The framework consists of two types of cryptographic modules, the asymmetric and the symmetric that assist in achieving the secure and fast communication as per the requirement of allow latency safety applications. Two communication protocol shave to been proposed that help in authenticating the credential as well as the platform in the communication medium. The simulation results showed that the proposed for framework satisfy the requirements for low latency safety applications. It performs substantially well in terms of some performance parameters; particularly for its latency in network. However it is noticed that, if nodes in the group are few and upcoming/joining nodes are higher than there is significant impact on the performance 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.

[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.