Indian Journal of Science and Technology, *Vol 8(S9), 300–305, May 2015*

ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645
DOI: 10.17485/ijst/2015/v8iS9/65574

Dynamic Data Fault Tolerance Mechanism to Enhance Reliability and Availability in Cloud

S. Giriesh, V. Sindhuja, P. Padmakumari and A. Umamakeswari

Department of Computer Science and Engineering, School of Computing, SASTRA University, Thanjavur - 613401, Tamil Nadu, India;
girieshs@gmail.com, 1994.sindhu@gmail.com, padmalec.sastra@cse.sastra.edu, aum@cse.sastra.edu

Abstract

Background/Objectives: The main objective of this project is to identify and overcome those frequently occurring cloud computing failures and in turn maintaining the integrity of the system. **Methods/Statistical analysis:** Nowadays, predominantly existing failures are component failure, network failure and security issues. A practical problem is considered, that occurs frequently in our system, and the results also confirm that the downtime can be shortened even if failure occurs. In this paper a new mechanism is introduced namely CFTM (Collaborative Fault Tolerant Mechanism) to overcome these failures in the cloud environment. Results: The CFTM provides with authentication to the users who upload their data into the cloud. By fragmenting and replicating them into the virtual storages, there is least chance of data loss in the server end even when there is a system component failure in the server side that will result in the data that are stored. Therefore, the CFTM (Collaborative Fault Tolerant Mechanism) provides the user with reliability, availability and throughput. Collaboration of authentication and data recovery is the key feature of CFTM. It makes sure that the integrity, reliability and the scalability of the data that the user handles over the cloud server are enhanced and maintained. The CFTM works greatly on a smaller scale meaning that it will deal with systems that are under a smaller sized network. Since the mechanism revolves around basic techniques of computing, this mechanism can be applied to any bigger network. **Conclusion/Application:** The Collaborative Fault Tolerance Mechanism is a successful mini scale fault tolerance mechanism that provides effective cloud service to the user. CFTM can be applied to any cloud related system.

Keywords: Availability, CFTM, Dynamic, Fault-Tolerance, Reliability

1. Introduction

Cloud computing relies on sharing computer resources rather than having servers in local for handling applications. Internet is the keyword for a cloud service, meaning the services are provided over the internet throughout the world. The massive storage area a cloud provides is prone to have several faults. These faults are likely to affect the performance of the cloud. Therefore, there is mandatory need to overcome these faults in order to restore the efficiency and reliability of the cloud. This paper focuses on the collaborative mechanism using which these faults can be resolved. Several approaches use the Virtualization technology (Fault Tolerance Manager) in order to migrate and replicate several virtual machine instances to achieve fault tolerance[1]. Existing anomaly management approaches for cloud environments can be commonly classified into two categories: reactive and proactive fault tolerance approaches and failure induction approaches[2]. Other existing mechanisms are: 1. Check pointing is a n efficient method by which, process states are saved at every change thereby making sure that on account of any failure, the system can toll back to its previous state instead of starting from the beginning. 2. Job migration is another method where a particular task that is causing a fault can be migrated to another system and brought back

**Author for correspondence*

to the current system. 3. Safety bag checks, where every task is assigned set of safety properties that are to be met[3]. For security reasons in cloud SecHDFS and SecCloud are used to achieve the security goals[4]. Data is fragmented based on privacy constraints, thus it uses BEA[5].

2. Concept of Fault Tolerance

The massive storage area a cloud provides is prone to have several faults. This paper will deal with three of the main faults any cloud service will have to overcome to provide a secure and reliable service to the customers. The main faults that this paper deals with here are authentication, network failure, and server side data loss. To depict the mentioned faults and recovery, faults are manually induced on a small scale cloud provider and the recovery is shown.

3. Collaborative Fault Tolerant Mechanism (CFTM)

Fault Tolerance is one of the key features to provide an efficient and reliable cloud environment since data is dealt with in huge amounts. A schematic sketch of the overall idea is shown below. Java coding language is used for the application development. The application is split up into four main modules 1. Authentication module, 2. Jar making module, 3. Fragmentation module and 4. Replication module along with retrieval.

Authentication module deals with the identity of an individual user. It makes sure that every user account is safe from intrusion. Jar making module deals with a local buffer area in the user side of the process which buffers the uploading data to the cloud storage. This module helps in reducing the upload time on the event of a network fault. Fragmentation module is present on the server side and handles the division of data that is being uploaded to the cloud by the user. Replication module along with retrieval module deals with storing the fragmented data into several available virtual servers present in the server side that is the cloud. In order to provide ease to the user to handle his personal data, a retrieval module is made to fetch the data that he has stored.

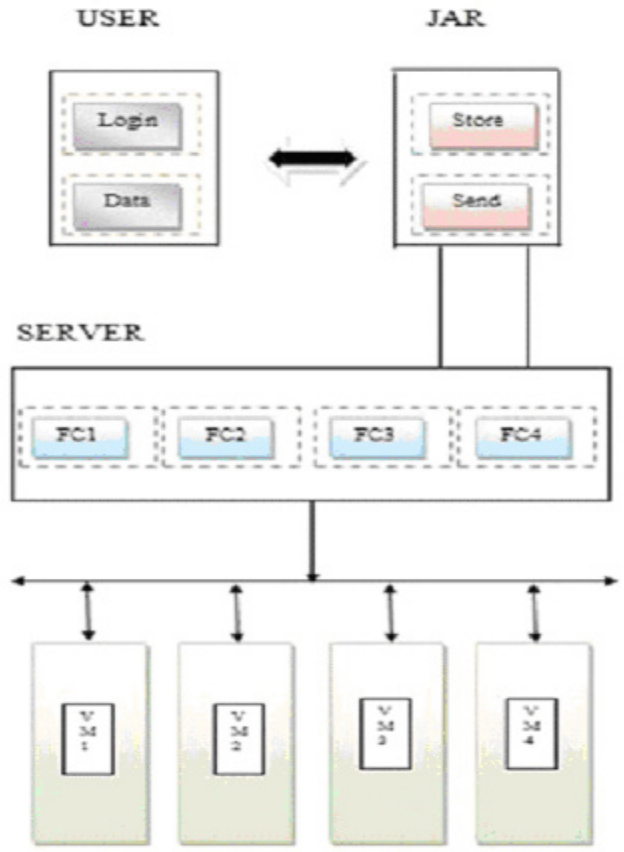

Figure 1. Overall Sketch of CTFM.

3.1 Authentication

The important sector in cloud computing is to provide complete security to the final users[6]. The first type of authentication is verifying a genuine identity. The second type of authentication is by comparing objects of that type. The third type of authentication is merely based on documentation. The main purpose of an Authentication module in the paper is to provide security to the user data. In order to provide such secure cloud storage, each user is allowed to create an account. This setup is similar to maintaining an account on Gmail. Once the user sign's up, his or her details are stored in the server database. But this step is common in any account based setup. It uses encryption of data for data confidentiality[7]. To differentiate this paper from others papers, a signature based authentication module is created. The approach is simple where SHA-Secure Hashing Algorithm is adapted. The idea is to generate a unique key for each user that is registering to the cloud service.

SHA1 Algorithm

1. Create object md of class Message Digest
2. data ←user Attributes
3. in←data
4. update(in.getBytes)
5. out ← digest
6. final ←bytesToHex(out)

The above algorithm describes the SHA1 (Secure Hashing Algorithm) for the generation of authenticated key. Initially the user attributes i.e. user name and the password are together stored in a variable (data). The variable data is stored to a string variable (in) which in turn is split into bytes by the get Bytes function call. The output of the digest function (takes care of padding) is stored to another string variable (out). The variable out is then passed as parameter to the function bytesToHex that converts the byte array into hexadecimal format. The output of the bytesToHex function is the generated key that will be used for user authentication during login. In the bytesToHex function, an character array is defined with the digits 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F.

3.2 Jar Making

The next module is the JAR making module which helps to recover from a client side fault. The JAR acts as a local buffer space which accumulates the data before it is being sent to the cloud so that, when there is an issue relating to the network, re uploading of data can be avoided saving data usage to the client[8]. The main idea behind this buffer area is that, in other file upload processes to a cloud, the conversion of data into transferable format totally relies upon a strong established network. That dependency is removed as the data that is stored in the buffer space is already in the transferable format. This makes the data ready for upload. One important point to be noted is that the speed of file upload from the user side to the cloud is generally very fast. Therefore, once the network recovers from the temporary downtime, the upload resumes immediately. To depict this, as mentioned earlier the error is induced manually. A switch that controls the network is available. When switched off, the network goes offline. Meanwhile, the data gets stored in the buffer. Changing the status of the network to online mode enables the user to send the data to the server from the buffer.

3.3 File Fragmentation

Fragmentation is a process of partitioning or representing a database where the database is split into several parts and then stored in units of different systems in a network. Data that has been broken down can be recollected as a whole. During fragmentation, there are certain conditions that are to be met for the fragmentation process to be effective and correct[9]. Completeness, Reconstruction and Disjointness are the three main points to be remembered when a data set is fragmented.

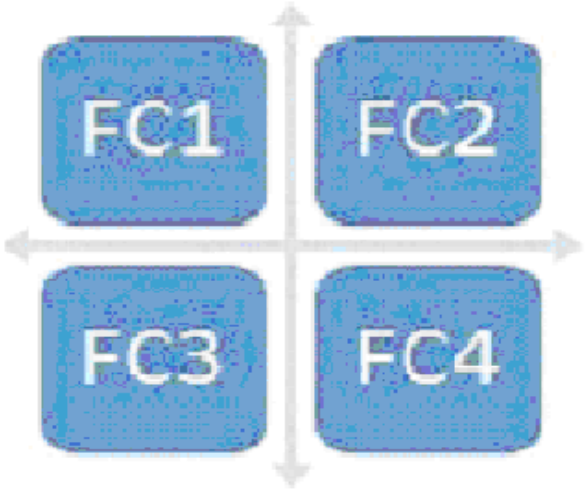

Figure 2. Fragmented Clusters.

The next module deals with the server side error. This fault is mainly handled by the service providers and has nothing to do with the clients, i.e. Clients are blind to the back end of the server. Generally, it is known that a cloud storage has several number of Virtual storages. When data is stored in cloud storage, data are sent to respective data locations in those virtual storages. When one of these Virtual storages goes down, then the fate of the data stored there is a big question mark. Although there are many mechanisms that provide data recovery, this paper deals with a different type of mechanism. The concept of file fragmentation is used. When a data file is uploaded to cloud storage, the file is split or fragmented into several parts depending upon the size of the file. The system configuration and the size of the file are taken into consideration and the fragmentation pattern is decided. Say, if there is a text file that contains 100 text characters and there are four virtual servers set up (Virtual servers are for data replication), then the split will be 25 characters per split. Fragmentation is not a separate process; it is followed by the replication module.

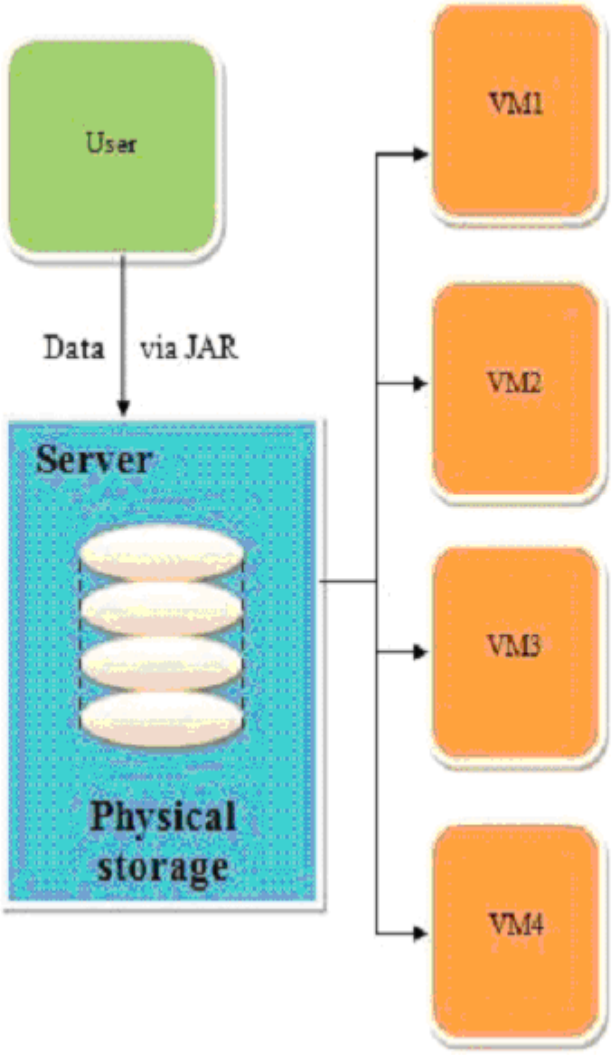

Figure 3. Physical storage to virtual machines.

3.4 Replication

Replication is a key strategy for improving reliability, availability and performance in distributed systems. Vital role of a file system is to store data reliably. Distributed File System (DFS) which is similar to a local based file system, stores files on one or more location called servers10. DFS ensures multiple copies of same file in different servers.

3.4.1 Reliability

DFS eliminates single point failure which helps the client to switch over to one of the replicated server when the current server becomes unavailable.

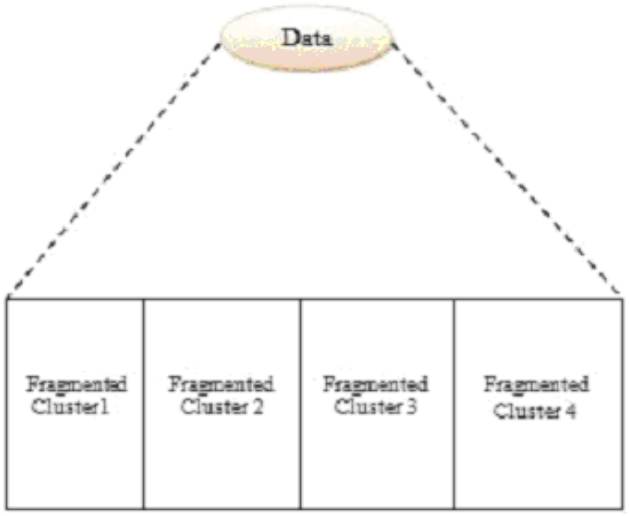

Figure 4. Each data into 4 Clusters.

3.4.2 Availability

DFS allows the files to be available for the client all the time. This is achieved by taking backups by the administrator while the system is up and running.

3.4.3 Fault Tolerance

Any system is prone to failure and there is a compulsory need to act to that failure in order to provide sustained service. This process is collectively termed as Fault tolerance. Here we have four virtual servers that have the fragmented files which are replicated over all the servers, in order to enhance reliability and availability.

3.4.4 Performance

DFS has the ability to make a platform to all variety of users to store their files and retrieve it later.

The basic idea of replication is to make replicas of same resource at different locations of servers. Client could access the cloud which takes the closest server and performs accordingly. This helps in reducing congestion in the network and load for server.

After fragmentation, the split files are replicated. By this way, it is assured that every virtual server will have a part of the data. Even when a virtual server is down, the data can be recovered from the rest of the virtual servers that are running. For example, say a file F is stored and say the size of the file is 100 bytes. Let there be 4 virtual servers inside the server. Since the size is 100, there will be 4 splits of the file of 25bytes each. Say F1, F2, F3, and F4

Table 1. Comparison of replication algorithms with CFTM

Algorithms	Availability	Reliability	Network traffic	Throughput	Limitation
APCP (Asynchronous Primary Copy Protocol)	More	More	Moderate	More	-
JRM (Junction Replication Method)	More	More	Less	More	Protection
Less log	More	More	Less	Less	Large scale P2P system
DORA (Dynamic file assignment strategy with replication)	More	More	Moderate	More	Read Dominant
OPR (Online replication algorithm)	More	More	Less	More	Complex workloads on live environments
OPR (Online replication algorithm)	More	More	Moderate	More	Large scale replication
Access – pattern and bandwidth aware replication	More	More	High	More	Locality of access Patterns
Plover (Proactive overhead replication method)	More	More	High	More	Efficient lookups
EDFRS (effective distributed file replication system)	More	More	High	More	-
CFTM (Collaborative Fault Tolerant Mechanism)	High	High	No	High	-

are the splits after fragmentation. Files F1, F2 and F3 will get stored in the virtual server 1. Files F2, F3, F4 will get stored in the virtual server 2. Files F3, F4, F1 will get stored in the virtual server 3. Files F4, F1, F2 will get stored in the virtual server 4. Therefore, if one of the virtual servers go down, data can be retrieved from the rest of the three virtual servers. The comparison of various replication algorithms are shown below along with CFTM[11].

4. Conclusion and Future Enhancement

This paper deals with a new fault tolerance mechanism namely the CFTM (Collaborative Fault Tolerant Mechanisms) that provide the user with reliability, availability and throughput. Collaboration of authentication and data recovery is the key feature of CFTM. It makes

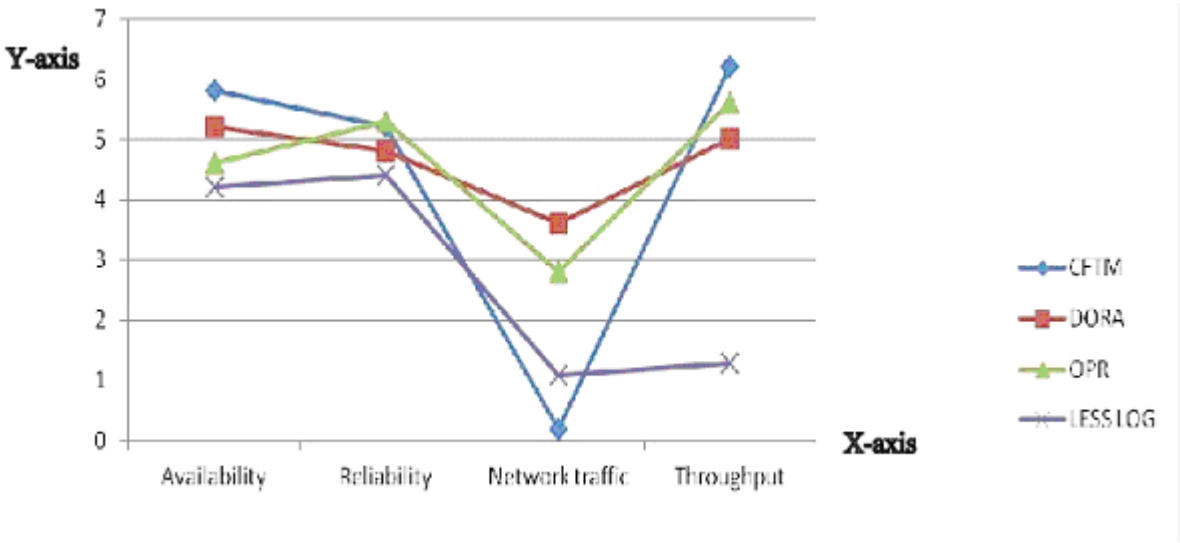

Figure 5. Comparison Graph.

sure that the integrity, reliability and the scalability of the data that the user handles over the cloud server are maintained. The mechanisms are implemented on a small scale level and can be scalable to a large scale level. Since the conversion of image and video data can be complex and large scale, this paper is limited to the upload of data in bytes i.e. text format. Future advancements of this paper can be the upload of videos and images to the cloud server. The CFTM can be applied to any large scale cloud providing system.

5. References

1. Jhawar R, Piuri V. Fault tolerance management in IaaS clouds. 2012 IEEE First AESS European Conference on Satellite Telecommunications (ESTEL); IEEE; 2012.
2. Abid A, Khemakhem MT, Marzouk S, Jemaa MB, Monteil T, Drira K. Toward antifragile cloud computing infrastructures. Procedia Computer Science. 2014; 32:850–5.
3. Randles M, Lamb D, Taleb-Bendiab A. A comparative study into distributed load balancing algorithms for cloud computing. 2010 IEEE 24th International Conference on Advanced Information Networking and Applications Workshops (WAINA); IEEE; 2010.
4. Wei L, Zhu H, Cao Z, Dong X, Jia W, Chen Y, et al. Security and privacy for storage and computation in cloud computing. Inform Sci. 2014; 258:371–86.
5. Shao Y, Shi Y, Li H. A novel cloud data fragmentation cluster-based privacy preserving mechanism. International Journal of Grid and Distributed Computing. 2014; 7(4):21–32.
6. Gharehchopogh FS, Bahari M. Evaluation of the data security methods in cloud computing Environments. IJFCST. 2013; 3(2):41–51.
7. Hudic A, Islam S, Kieseberg P, Weippl ER. Data confidentiality using fragmentation in cloud computing. IJCNDS. 2012; 1(3/4):1.
8. Sundareswaran S, Squicciarini AC, Lin D. Ensuring distributed accountability for data sharing in the cloud. IEEE Transactions on Dependable and Secure Computing. 2012; 9(4):556–68.
9. Navaz A, Prabhadevi C, Sangeetha V. Data grid concepts for data security in distributed computing. 2013. arXiv preprint. arXiv:13086058.
10. Stockinger H, Samar A, Holtman K, Allcock B, Foster I, Tierney B. File and object replication in data grids. Cluster Comput. 2002; 5(3):305–14.
11. Ciciani B, Dias DM, Yu PS. Analysis of replication in distributed database systems. IEEE Trans Knowl Data Eng. 1990; 2(2):247–61.