IJCTA, 9(10), 2016, pp. 4277-4283

Live Migration Approach of Fault Tolerance in Cloud Infrastructure Using LMMC and Sbaalgorithsm

***S. Indirani **Dr. C. Jothi Venkateswaran**

Abstract : The ever increasing demand and advantages of cloud computing infrastructure has led to real time computing being performed on cloud infrastructure. On the other hand, many of the real time systems have to be also critically safe and must be greatly robust. As a result, several researches were performed to implement fault tolerance in the distributed systems, within which live migration for the purpose for fault tolerance has a considerable role to play. But, some researches on the fault tolerant side have adequate study conducted on the Live Migration and virtualization, which are the two key features of clouds. In order to deal with these issues, the proposed algorithm presents a method for Live Migration Approach of Fault Tolerance against of VM and also reliability during performance variations of Cloud resources. Location based Minimum Migration in Cloud (LMMC) proposed for monitoring the overloaded tasks from VM thereby to enhance the utilization of resource and prevent faults. Fault-tolerant mechanism that extends the Seed Block Algorithm (SBA) model is used for incorporating the cloud features. The SBA is utilized for backup and recovery of tasks, and simultaneously checks the number of tasks running and the number of tasks it is capable of running.

Keywords : Cloud Infrastructure, Live Migration and Virtualization, Location based Minimum Migration in Cloud (LMMC), Seed Block Algorithm (SBA)

1. INTRODUCTION

Cloud computing can be described as an internet based computing solution which is regarded as one forward step in the technological development of distributed computing. It yields an extensive solution for the delivery of IT in the form of a service and it enables scalable and cooperative sharing of resources between various organizations. The cloud facilitates on demand access to applications from any place in the world, without any consideration towards their details of implementation [1].

Storage, processors, platforms, or application services comprise the resources. The flexibility of cloud computing is basically a function of assigning resources on demand [2], [3]. In clouds, resources of various physical machines can be categorized into Virtual Machines (VMs). These VMs can start and stop on-demand in order to meet service requests. This renders maximum flexibility for the purpose of configuring different partitions of resources for certain multiple requirements of user requests [4].

VMs can fall short in doing their work owing to their heterogeneity and utilization over longer periods of time. Failure of VMs strikes a huge effect on the scalability, performance, profit and consumer trust. Fault tolerance is an approach in which a cloud computing system persists to operate successfully even in the faulty conditions [5].

Usage of cloud infrastructure for real time applications maximizes the risks of errors, since the cloud nodes (virtual machines) are far away from the transceiver (job submitting node, actuator or sensor). Several real time

* Research Scholar, Mother Teresa Women's University, Kodaikanal-624 1022

** Head of the Department, Department of Computer Science, Presidency College, Chennai – 600005. jothivenkateswaran@yahoo.co.in.

systems are also safety critical systems in addition, hence a higher degree of fault tolerance is necessary. Safety critical real time systems need to operate properly in order to avoid failure, which can lead to financial loss and casualties as well [6]. Hence there is an ever increasing necessity for fault tolerance for such kind of systems when used with cloud infrastructure.

This paper is organized as follows. Section II describes the literature review on fault tolerance.Section III presents proposed algorithms mechanism and Section IV presents conclusions and future work.

2. LITERATURE REVIEW

Scheduling is an effective approach for gaining fault tolerance through the allocation of multiple copies of tasks on different computing instances. Among the researchers conducted on fault-tolerant scheduling, the Primary-Backup (PB) model is typically a well-known scheme where every task contains two copies, which are, the primary and also the backup, doing the execution on two different kinds of computing instances for fault tolerance [7].

Scheduling is used for distributing resources among parties which simultaneously and asynchronously request them. Scheduling algorithms are used mainly to minimize resource starvation and to ensure fairness amongst the parties utilizing the resources. Scheduling deals with the problem of which resources needed to be allocated for the received task. With Scheduled Tasks, can schedule any script, program, or document to run at a time that to specify when creating the task. The task scheduling problem optimizes the overall performance of the application and provides assurance for the correctness of the result[8].

Comprehensive researches have been carried out in order to design effective fault-tolerant scheduling algorithms in accordance with the PB model used in the conventional distributed systems like grids, clusters [9]. But, while differing from the age old distributed systems, clouds have few features that are distinct: One of the significant technologies in cloud computing is virtualization that causes VMs to become the fundamental computing instances, and helps VMs to move across the multiple-hosts; The scale of cloud is set to elastic as per the demands of the users. It can also be scaled until to meet up with the increasing resource allocation requests when also scaled down for improving the utilization of resources with the demand being comparatively low. The features above suffer more hardships for the research in clouds.

In an attempt to get a solution to this problem, Fault-tolerant Elastic Scheduling algorithms for real-time TAsks in cLouds (FESTAL) is proposed [10].Virtualization and the elasticity are taken to be in need of highly efficient fault tolerant scheduling for using in real-time tasks in the case of virtualized clouds, and then is developed FESTAL, *i.e*., Fault-tolerant Elastic Scheduling algorithms for real-time Tasks in clouds. Based on the fault-tolerant mechanism and the elastic resource provisioning mechanism, design of an effective fault-tolerant scheduling algorithm – FESTAL is done. The chief setbacks of this FESTAL fault-tolerant mechanism are that it cannot handle the failure of multiple hosts. The performance of FESTAL is low when operating with multiple hosts. The existing system only takes scheduling and fault identification into consideration in order to improve the accuracy of the model.

3. PROPOSED LMMC AND SBA MECHANISM

In this paper, the combination of Location based Minimum Migration in Cloud (LMMC) and Seed Block Algorithm (SBA) is used for effective Scheduling in Cloud, and smart remote data backup algorithm is employed for fault tolerance. Fig.1 shows system architecture of proposed system. The cloud service generally runs in data centers. The vital goal of a data center is to host the applications that can manage the core business logic and then process the operational data of the organization. Presently, data centers may have hundreds of thousands of servers.

The common elements can be databases, file servers, application servers, middle-ware, and several others. A multi-tier *e*-commerce application may comprise of web servers, database servers and application servers. The front end web server provides the display of information like merchandise lists. The application server communicates with the front end web server and realizes the core functionality of the application. The database server maintains the transaction data and keeps it neutrality and is independent of application logic.

A key enabling technology for the case of cloud computing is virtualization, that enables the abstraction of the physical infrastructure, conceals the complexity behind the underlying hardware or software, and keeps the infrastructure available as a soft component which is easy for isolation and hence share physical resources. As the data centers keep on deploying virtualized services, there are several scenarios requiring migrating VMs from one physical machine to another located within the same data center or across multiple data centers.

In order to maintain high performance and availability, VMs require to be migrated from one cloud to another in order to supplement for better resource availability, minimize down time due to hardware maintenance, and get over computing power limitation in the source cloud. This way VMs may be relocated across different physical machines in a cluster for the purpose of relieving workload on hosts affected by congestion.

The increase in workload of a virtual server can be dealt with by maximizing the resources assigned to the virtual server with the condition that some of the idle resources are generally available on the physical server, or by simply migrating the virtual server to a physical server that is loaded less. The use of new SBA is double; first SBA aids the users to collect data from any remote location during the times of network connectivity absence and secondly, for the recovery of the files if the file deletion or else the cloud is destructed because of some reason.

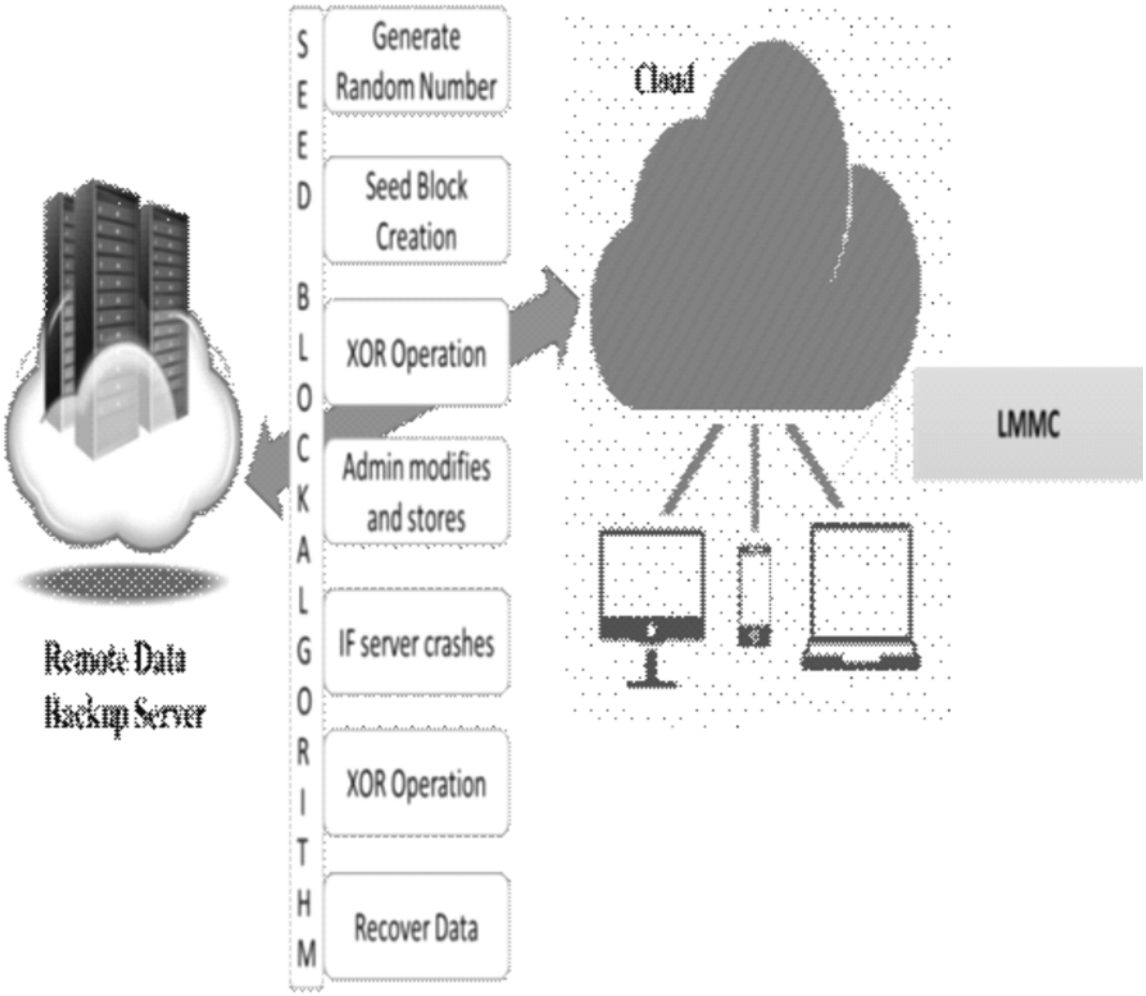

Fig. 1. System Architecture of our proposed system.

A. Location based Minimum Migration in Cloud (LMMC) Algorithm

The overlapping technique helps in the improvement of the resource utilization by minimizing the backup's utilization of the processor time available, which however operates only at the VM level. More improvement is supposed to be gained at the host level. Then the VM migration, a commonly employed technique in the cloud

resource management, is capable of further optimization of the utilization of the resource at the host level. In this paper a novel method is proposed which is referred to as Location based Minimum Migration in Cloud (LMMC) and also does a check on the number of tasks that running and number of tasks that are able to be run.

The resource assumption is made which means there is a necessity to conclude on the number of resources and the resources configuration. In order to evaluate the task, the task from the user is examined and forwarded for the allocation process. The resources are then allocated for the jobs given by the users. The provisioning of these computation resources is managed by a provider, and resources are assigned in an elastic way, as per the consumer's needs. For supporting the accommodation of unforeseen demands on the infrastructure in a scalable and elastic way, the procedure of allocation and reallocation in Cloud Computing must be dynamic.

The trust levels are provided for all resources and the user will give the task along with the security demands for running the task with the resources. On the basis of the user demanded security, the resources are then allotted for each of the user jobs. Model analysis takes place for migration process, where the decision on the tasks to be migrated is made. Location based VM selection policy (algorithm) has to be employed to conduct the selection process. The tasks are migrated on the basis of the analysis of the last process. When a virtual machine is observed to be under loaded (minimum task running) needing live migration to one or more VMs (which are already running some jobs) from the neighboring location cloud centers.

Location based VM selection policy (algorithm) has to be employed to conduct the selection process. The important contributions are given below, Migration of Virtual machine tasks where less number of tasks are running, that occupies additional energy and power consumption. Migration of Virtual machine tasks where number tasks are running, is an overhead to the system performance. Less number of migrations should be chosen, to minimize more processing cost for migration process. In VM migration, a virtual machine is transferred from one host machine to another host machine lively without any disruption on the application running on it. Utilizing virtual machine migration accomplishes:

1. Load balancing between the resources through distribution of overall workload.
2. Reduce power consumption through optimization of resource utilization of tasks.
3. Prevent failure and improve the availability of the resources.

Employing a load balancing migration technique helps in distributing the load across the VMs in a data center. Dynamic workload in cloud data center is dealt with the live migration of VMs. The Load balancing with VM migration yields advantages such as, improvement of resource utilization, minimization of migration cost, enhancement of scalability, prevention of the over provisioning of resources.

B. LMMC Scheduling Algorithm

function LMMC schedule

Input : Input task (t_i), $H_a = \{h_1, h_2, \ldots, h_{j\,|Ha}\}$, Number of vms ($V_i$), Cloud Resources (R), arrival time *ai*, deadline *di*, and task size *tsi*,

Output : Best Resource (R2)

Initialize : R – 0, temparr, tempp, arrtime

Temptrl – 1000, rtask – t_i, arrtime-*ai*, deadtime – *di*

for $h = 0 $h < $hostincr $h

for $i = 0 $i < $vmsincr $i

if $ temparr[$i] = 0 then

$tempp = $clres[$i];

$ arrtime = $*ai*[$i];

$ deadtime = $*di*[$i];

if $rtask < = $tempp & arrtime& deadtime then

if $rtask< = $task size then

```
if                          $temptrl  >  $tempp then
                            $temptrl  =  $tempp;
                                $R    =  $i;
end if
end if
end if
if                          $temptrl  =  0 then report  "Cloud Space not available"
else                      $clres[$R]  =  $rtask
                        $temparr[$R]  =  1 report "Cloud Resource Allocated"
                                R2    =  R + 1;  return R2;
LMMC Migration Algorithm  functionLMMCmigrate
Input : Number of  vms (vms), Loaded Cloud Resources (load1)
Output : Migration
Initialize load 2, hvms, n load, t1 0, t2 0
for                             $i    =  0 $i < vmsincr $iload2 [i] = load1[i];
for                             $j    =  0 $j < vmsincr $j
if                        $load2[$j]  <  4 &&  $load2 [$j]  > 0 then
                         $hvms[$t1]   =  $j; $t1+ +;
end  if
for                             $j    =  0 $j < vmsincr $j
if                        $load2[$j]  <  4 && $ load2[$j] > 0 then
                          $nload[$j]  =  load2[$j]
else
if                              $t2   <  $t1 then
                                $k    =  $hvms[$t2]
                               $v1    =  $load2[$k]
if                  $load2[$j]+$v1<   =  10 &&$ load2[$j] > 0
                          $loa2[$k]   =  0
$load2                      [$j]    =  $load2[$j] + $v1
                         $nload[$k]   =  $load2[$k]
                          $nload[$j]  =  $load2[$j] $t2 + +;
else                      $nload[$j]  =  $load2[$j];
else                      $nload[$j]  =  $load2[$j];  return  $nload
```

C. Seed Block Algorithm

This algorithm is focused on simplifying the backup and recovery procedure.Fig.2 shows architecture of Seed Block Algorithm. It normally makes use of the idea of Exclusive–OR (XOR) operation of the world of computing. For ex : – Consider there exists two data files: A and B. When XOR A and B is done, it produces X *i.e*. X = A (XOR) B. Suppose A data file is destroyed and A data file is needed back then it will be capable of getting A data file back. It is easier to retrieve with the support of B and X data files *i.e*. A = X (XOR) B. In a similar manner, the Seed Block Algorithm operates in a way to render the simpler Back-up and recovery procedure. The important goal of the remote backup service is helping the user to gather data from any remote location even in the absence of network connectivity or if the data not observed on main cloud. If the data is not seen on central repository, clients are permitted access to the files from remote repository (*i.e*. indirectly). The architecture of this algorithm is

given in Fig.2. Fig.2 comprises of the Main Cloud and its clients and the Remote Server. Here,a random number is first set in the cloud and unique client id is set for each client. Secondly, everytime the client id gets registered in the main cloud; then the client id and the random number get EXORed (XOR) with each other in order to generate the seed block for the concerned client. The seed block generated corresponding to every client is maintained in the remote server. Whenever a client generates the file in the cloud for the first time, it gets saved in the main cloud. When saved in the main server, the client's main file is EXORed with the Seed Block of that particular client. Then that EXORed file gets saved in the remote server as a file (referred to as File dash). In case, unluckily if a file in the main cloud gets crashed / destroyed or the file is deleted by mistaken, then the user shall get back the original file through EXORing the file' with the seed block of the respective client to generate the original file and renders the result file *i.e*. the original file to the client who requested it. Each task ti has two copies represented as t_i^O and that are executed on two different hosts for the purpose of fault tolerance.

SBA Algorithm process steps

The SBA algorithm proposed is as below :

Initialization : Main Cloud: M; Remote Server : Rs

Clients of Main Cloud : Files : and a_1 and a'_1 See Block : Si ; Random Number : r ; Client's ID : Client_I*di*

Input : a_1 created by ; r is generated at M*c* ;

Output : Recovered File a_1 after the deletion at M*c* ;

Given : Authenticated Clients can permit uploading, downloading and modify its own files only.

Step 1 : Generate a random number. int $r = r$ and( );

Step 2 : Create a Seed Block Si for each Ci and store Si at, = $r \oplus$ Client_Idi (Repeat step 2 for every client)

Step 3: If Ci/ Admin creates /modifies a a_1 and Stores at, then a'_1 create $a'_1 = a1 \oplus$ Si

t_i^B

Step 4 : Store a'_1 at Rs

Step 5 : If Server crashes a_1 deleted from M*c*, then EXOR is done to get back the original a_1 as : $a_1 = a'_1 \oplus$ Si

Step 6: Return a_1 to C*i*

Step 7: END

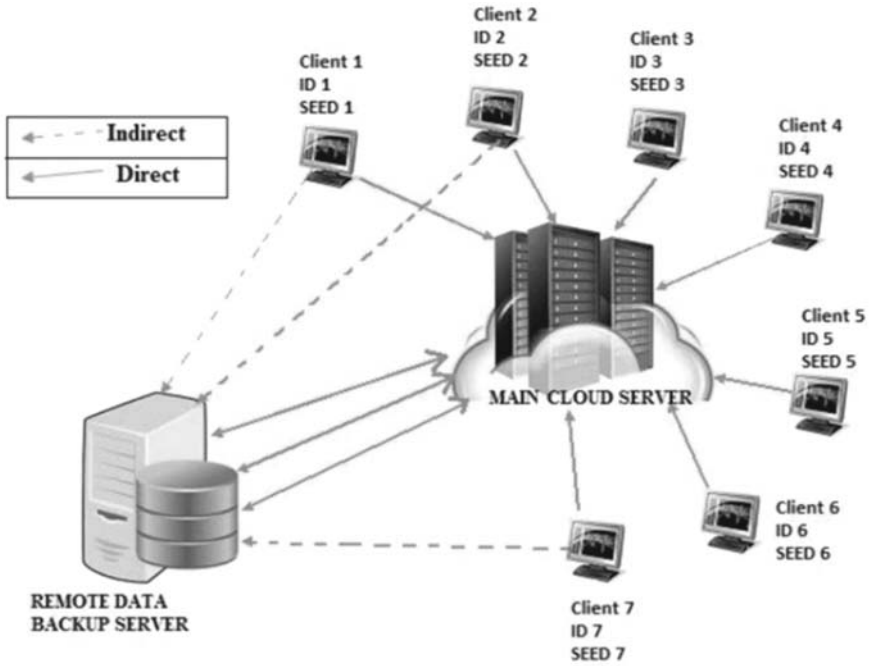

Fig. 2. Architecture of seed block algorithm.

4. CONCLUSION AND FUTURE WORK

Cloud computing can be defined as two applications that are delivered as services over the Internet and the hardware and systems software present in the datacenters providing those services. Failures of any kind are common in the datacenters in the recent times, partially because of the number of nodes. The virtualization technology in data centers has helped in the improvement server utilization and server consolidation when minimizing the resource management complexity and costs. The proposed scheme is a good choice for use as a fault tolerance mechanism for the purpose of real time computing on cloud infrastructure.

Frequently occurring of VM outage incidents points out that fault tolerant techniques of VM are necessary for achieving high availability of cloud computing infrastructure.In future implementation will be doneascombination of LMMC and SBA algorithms in cloud environment using simulation tools in terms of improving power savings, execution time and less cost.

REFERENCES

1. Alhosban, A., Hashmi, K., Malik, Z., & Medjahed, B. (2013) "Self-healing framework for Cloud-based services", ACS International Conference on Computer Systems and Applications (AICCSA), 1-7.
2. Das, P., & Khilar, PM (2013) "VFT: A virtualization and fault tolerance approach for cloud computing", Proceedings of the IEEE Conference on Information and Communication Technologies (ICT), 473-478.
3. Ganga, K., & Karthik, S., (2013) "A fault tolerent approach in scientific workflow systems based on cloud computing", International Conference on Pattern Recognition, Informatics and Mobile Engineering (PRIME), 387-390.
4. Buyya, R., Yeo, CS., Venugopal, S., Broberg, J., & Brandic, I., (2009) "Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility", Future Generation computer systems, 25 (6), 599-616.
5. Singh, D., Singh, J., & Chhabra, A., (2012) "High Availability of Clouds: Failover Strategies for Cloud Computing using Integrated Check pointing Algorithms", IEEE In Communication Systems and Network Technologies (CSNT), 698-703.
6. Reis, GA., Chang J., Vachharajani, N., Rangan, R., & August, DI., (2005) "SWIFT: Software implemented fault tolerance", In Proceedings of the international symposium on Code generation and optimization, IEEE Computer Society, 243-254.
7. Ghosh, S., Melhem, R., & Moss´e, D., (1997) "Fault-tolerance through scheduling of aperiodic tasks in hard real-time multiprocessor systems" , IEEE Transactions on Parallel and Distributed Systems, 8(3), 272–284.
8. Balasangameshwara, J., & Raju, N., (2013) "Performance-driven load balancing with a primary-backup approach for computational grids with low communication cost and replication cost," IEEE Transactions on Computers, 62(5), 990–1003,
9. Ravichandran, S., Naganathan, E.R., (2013) "Dynamic Scheduling of Data Using Genetic Algorithm in Cloud Computing", International Journal of Computing Algorithm, 2(1), 127-128.
10. Wang, J., Bao, W., Zhu, X., Yang, L. T., & Xiang, Y., (2015) "FESTAL: Fault-Tolerant Elastic Scheduling Algorithm for Real-Time Tasks in Virtualized Clouds", IEEE Transactions on Computers, 64(9), 2545-2558.