A Study on Fault Tolerance Solution

Dr. J Meena Kumari
Professor & Head
Computer Science Department,
Oxford College of Science, Bangalore, India.

Shaima'a, Ghamdan
Research Scholar,
Computer Science Department,
Jain niversity,Bangalore,India.

Abstract: **Cloud Computing has gained popularity in the recent years. Node failure when applications run in servers are a major concern to be addressed. This is otherwise called as Fault Tolerance. Reliability and availability of nodes during execution of critical service applications is a major concern which may otherwise affect the quality of service provided by the cloud service providers.**
In order to reduce the impact of failure when an application runs on the cloud, there should be mechanism to anticipate the failures so that failures can be proactively addressed. This paper addresses the various fault tolerance techniques used to deal with various software faults in a virtualized cloud environment. Fault-tolerance is an important issue in job scheduling on cloud data centers. One way of providing fault-tolerance is to schedule multiple copies of a task on different virtual machines. Replication is of the most common methods in fault tolerance but still it faces lots of problems. This paper discusses different techniques of fault tolerance and highlights the problems met proactive and replication as one of these techniques.

Keywords- Fault tolerance, real time system, virtual machine, proactive and replication.

1. INTRODUCTON:

Computer network technologies have improved in the last 20 years. After the arrival of Internet the networking of computers has led to several novel advancements in computing technologies like Distributed Computing and Cloud Computing. The term distributed systems and cloud computing systems refer to different things, however the underlying concept between them is same.
Companies outsource their IT services to third party providers due to the high cost of maintaining the internal infrastructure. This trend led to the emergence of the so-called cloud computing approach. These providers are called as Cloud Service Providers. One of the greatest advantages of cloud computing is to allow customers to pay only for the amount of resources used by them. The cloud provider is responsible for the administration of the cloud resources that includes hardware and virtual machines (VM) and services. It is the responsibility of the provider to manage the accommodation of the capacity of the cloud. Cloud customers make use of the resources provided by the cloud to deploy and execute their applications.
When real time applications of customers run in cloud infrastructure the chance of node failure is quite high. As most of the applications which run on cloud are safety critical systems. In general, real-time system is one that should process information and create a response within a scheduled time else may risk severe consequences including failure [12]. So the reliability depends not only on the logical result, but also the time of delivery [26]. Failure to respond when a system fails is equal to a wrong response [17]. The two characteristics which decide the reliability of cloud systems are timeliness and fault tolerance.
Fault tolerance is a concept that widely used in computer to indicate that the system continues working in the existence of failures. It has gained more concern with the emergence of cloud computing. With the large-scale and dynamic environment in clouds, errors become more popular and the need for fault tolerance becomes a must. The consequences of missing a deadline while scheduling jobs can be catastrophic in resource scheduling in cloud data centers in which the consequences are relatively tolerable.
In cloud computing there are two types of fault tolerance reactive and proactive. Reactive fault tolerance means to remove the fault after it occurs. Basically reactive fault tolerance reduces the effects of error on application execution. Proactive fault tolerance refers to avoiding faults, errors and failures by predicting them in advance.
In order to make the execution succeed, different replicas of task are run on different resources until the whole replicated task is accomplished. Replication is one of the most techniques used in clouds and considered as a reactive technique to reduce the impact of failure in the system. Proactive FT keeps an application alive by avoiding experiencing failures through preventative measures. It anticipates the failure using Pre-fault indicators, such as a significant increase in heat, can be used to avoid an imminent failure through anticipation and reconfiguration.

2. LITERATURE REVIEW

Cloud computing differs from standard distributed system since it is a real time system. Malik and Fabrice in [21] define real-time system as any information processing system which has to respond to externally generated input stimuli, within a finite and specified period of time [12]. So the correctness depends not only on the logical result, but also the time it was delivered [26]. Failure to respond is as bad as the wrong response [15][17]. These systems have two main characteristics by which they are separated by

other general-purpose systems. These characteristics are timeliness and fault tolerance [36]. By timeliness, we mean that each task in real time system has a time limit in which it has to finish its execution. And by fault tolerance means that it should continue to operate under fault presence [6].

Cloud computing has gained popularity because of its efficient way of referring to the use of shared computing resources. When several instances of an application running on several virtual machines, a problem that arises is implementation of an autonomic fault tolerance technique to handle a server failure to reassure system reliability and availability.

Proactive FT keeps an application alive by avoiding experiencing failures through preventative measures. In contrast, reactive FT keeps an application running through recovery from experienced failures. More precisely, proactive techniques anticipate, while reactive mechanisms respond. Although reactive solutions perform preventative measures, they do not avoid failures; instead, applications experience failures and recover [34].

In a dynamic cloud environment it is important that tasks complete within their deadline even in the presence of a virtual machine failure. There are three layers identifiable in a cloud platform: resources, VMs and applications. Each of them is concerned with failures. Therefore, we identify three types of failure in a cloud platform: hardware failure, VM failure and application failure.

3.1. FAULT TOLERANCE

In general, a failure represents the condition in which the system deviates from fulfilling its intended functionality or the expected behavior. A failure happens due to an error; that is, due to reaching an invalid system state. While fault tolerance, is the ability of the system to perform its function even in the presence of failures [14].

Fault tolerance is a setup or configuration that prevents a computer or network device from failing in the event of an unexpected problem or error [13]. It makes to achieve system dependability. Dependability is related to some QoS aspects provided by the system, it includes the attributes like reliability, safety and availability [4].

Definition of Fault Tolerance according to J. Ravi and P. Vincenzo [14] is the ability of the system to perform its function even in the presence of failures. While N. Toan et. al [25] defined fault-tolerance as a generic term that has long been used to name the ability of systems and applications to handle errors.

X. Kong et. al.[19] presented a model for virtual infrastructure performance and fault tolerance. But it is not well suited for the fault tolerance of real time cloud applications.

A VM failure is detected when the VM does not respond to a ping request or when the hypervisor indicates an error state. The research work in [33] focuses on fault tolerance in cloud computing platforms and more precisely on autonomic repair in case of faults. It discusses the implications of this splitting in the implementation of fault tolerance. In most of current approaches, fault tolerance is exclusively handled by the provider or the customer, which leads to partial or inefficient solutions. Solutions, which involve collaboration between the provider and the customer, are much promising.

In [14] the authors focus on characterizing the recurrent failures in a typical cloud computing environment, analyzing the effects of failures on user's applications, and surveying fault tolerance solutions corresponding to each class of failures. The perspective of offering fault tolerance as, a service to user's applications, as one of the effective means to address user's reliability and availability concerns is discussed. The most widely adopted methods in this approach is to achieve fault tolerance against crash faults and byzantine. Using the FTM framework, the notion of providing fault tolerance as a service can therefore be effectively realized for the Cloud computing environment.

The main advantages of using fault tolerance in cloud computing includes failure recovery, lower costs, and improved standards in performance [10], [1]. The notion of faults, errors and failures is understood as a chain [14] using the following definitions [5], [2]:

Fault =>Error => Failure => Fault=> Error=> Failure

3.2. PRACTIVE FAULT TOLERANCE

The notion of proactive fault tolerance emerged in recent years. It is a concept that prevents compute node failures from impacting running parallel applications by preemptively migrating parts of an application (task, process, or virtual machine) away from nodes that are about to fail. Pre-fault indicators, such as a significant increase in heat, can be used to avoid an imminent failure through anticipation and reconfiguration. Since avoiding a failure through preemptive migration is significantly more efficient than recovery from failure via traditional reactive FT mechanisms, such as checkpoint/restart, HPC system utilization becomes more efficient. [34].

There is a need for standardized metrics and interfaces. Currently, each solution uses its own metrics for measuring and evaluating health and interfaces between components. This makes comparison and integration unnecessary difficult. Another challenge is the need for reliable analysis to factor in performance parameters as the goal is to improve time to solution. More extensive work in performability analysis for HPC is needed [34].

3.3. FAULT TOLERANCE TECHNIQUE IN CLOUDS

The existing fault tolerance technique in cloud computing considers various parameters: throughput, response-time, scalability, performance, availability, usability, reliability, security and associated over-head as mentioned in [3][14]. A large number of studies have conducted on the data fault-tolerant systems in the recent years that resulted in the invention of new strategies for finding the advantages and obstacles of fault-tolerant systems. In this section, we introduce the most recent fault-tolerance techniques in the Cloud Computation. The main limitation is we cannot tolerate what we don't expect.

Fault masking: The fault masking is a structural redundancy technique to correct faults immediately for any kind of hardware redundancy. It completely masks the set of redundant modules. A set of similar modules execute the same functions and the output is voted to remove errors

created by a faulty module called as error voting. The drawback of this approach is that if faults are only hidden and fault detection is not used, the faulty components will not be detected, the available redundancy is going to decrease and the system will not be aware of that.

Reconfiguration: The ability of a system to alter the active interconnection among modules has a history of different purposes and strategies. Its purposes develop from the relatively simple desire to formalize procedures that all processes have in common to reconfiguration for the improvement of fault-tolerance, to reconfiguration for performance enhancement, either through the simple maximizing of system use or by sophisticated notions of wedding topology to the specific needs of a given process. The main drawback of this approach is that it discards the nominal controller completely in favor of a redesigned optimal controller. Therefore this approach is limited to systems which used an optimal controller or at least an observer/state feedback controller.

Check pointing–It is an efficient task level fault-tolerance technique for long running and big applications .In this scenario after doing every change in system a check pointing is done. When a task fails, rather than from the beginning it is allowed to be restarted that job from the recently checked pointed state.

Job Migration –Some time it happened that due to some reason a job can- not be completely executed on a particular machine. At the time of failure of any task, task can be migrated to another machine. Using HA-Proxy job migration can be implemented.

Replication- It is one of the most significant fault-tolerant techniques in storage centers that widely used in laboratory settings and in online service systems. Replication means "to copy." Different tasks are replicated for successful execution and optimal results, so replication performs on different resources. Replication can be executed through HA-Proxy, Hadoop and AmazonEC2.

Self-Healing- A big task can divided into parts .This Multiplications done for better performance. When various instances of an application are running on various virtual machines, it automatically handles failure of application instances.

Safety-bag checks: In this case the blocking of commands is done which are not meeting the safety properties.

S-Guard- It is less turbulent to normal stream processing. S-Guard is based on rollback recovery. S-Guard can be implemented in Hadoop, Amazon EC2.

Retry- In this case we implement a task again and gain. It is the simplest technique that retries the failed task on the same resource.

Task Resubmission- A job may fail now whenever a failed task is detected, In this case at runtime the task is resubmitted either to the same or to a different resource for execution.

Nowadays, demands for high fault tolerance and high serviceability are becoming unprecedentedly strong, building a high fault tolerance and high serviceability cloud is a critical, challenging, and urgently required task [40], [38]. Cloud serviceability and fault tolerance are still far from perfect. Failures are normal rather than exceptional in cloud computing environments, due to large-scale time-critical data support, and because cloud platforms are usually run in the form of voluntary.

3.4. REPLICATION AS FAULT TOLERANCE

One of the most widely adopted methods in [14] [18] to achieve fault tolerance against crash faults is replication. Replication: Critical system components are duplicated using additional hardware, software and network resources in such a way that a copy of the critical components is available even after a failure happens. Replication according to [10] is one of the most significant fault-tolerant techniques in storage centers. Replication means "to copy." Different tasks are replicated for successful execution and optimal results, so replication performs on different resources. Replication can be executed through HA-Proxy, Hadoop and AmazonEC2.

The basic mechanism to achieve the fault tolerance suggested by M. Sheheryar and F. Huet. [21] is replication or redundancy [6]. In [16] Replication: Various task replicas are run on different resources, for the execution to succeed till the entire replicated task is not crashed.

Due to the replication, cost for renting the cloud resources will increase. But it is really required to avoid the catastrophic loss [33].

In order to achieve high fault tolerance in clouds [32], in the replication fault tolerance strategy, there are three important problems that must be solved: which considers which data to replicate and when to replicate, and how many and where the new replicas need to be placed, and the trade-off between high fault tolerance and high cloud SLOs. With the number of new replicas increasing, the system maintenance cost will significantly increase, and too many replicas may not increase the availability, but cause unnecessary spending instead.

In [32] data replication mechanism in distributed computing can be classified into two groups: static mechanisms [8],[31]where the replication strategy is predetermined and well defined while dynamic replication mechanisms [39], [35],[37] automatically creates and deletes replicas according to changing access patterns. The major Fault tolerance (FT) strategies based on J. Ravi and V. Piuri in [14] can be divided into passive replication strategies and active replication strategies.

3.5. REPLICAION TYPES

There are two strategies for replication: active and passive. In active replication, all the replicas are simultaneously invoked and each replica processes the same request at the same time. This implies that all the replicas have the same system state at any given point of time and it can continue to deliver its service even in case of a single replica failure. In passive replication, only one processing unit (the primary replica) processes the requests while the backup replicas only save the system state during normal execution periods. Backup replicas take over the execution process only when the primary replica fails.

a. Primary-backup replication: In the primary-backup strategy [24], one of the replicas, called the primary receives the invocations from the client process, and sends

the response back. Given an object x, its primary replica is noted prim(x). The other replicas are called the backups. The backups interact with the primary, and do not interact directly with the client process.

b. Active replication: In the active replication technique, also called state-machine approach" [7], all replicas play the same role: there is here no centralized control, as in the primary backup techniques.

c. Passive Replication: In **passive replication** there is only one server (called primary) that processes client requests. After processing a request, the primary server updates the state on the other (backup) servers and sends back the response to the client. If the primary server fails, one of the backup servers takes its place. Passive replication may be used even for non-deterministic processes. The disadvantage of passive replication compared to active is that in case of failure the response is delayed.

Virtual machine replication (VM replication) is a type of VM protection and disaster recovery for all virtual machines in the environment. That takes a copy of the VM as it is right now and copies it to another VM. Replication of VM can be stored on the same host, different local host, or even a remote host.

4. ANALYSIS

Most of the real time applications require the fault tolerance capability to be provided. A lot of work has been done in the area of fault tolerance for standard real time systems. But there is lot of research room available in fault tolerance of real time application running on cloud infrastructure.

The cloud service provider usually host fewer physical servers and increase number of virtual machines (VMs). IT professionals must balance VMs with available resources. Virtualization saves costs by reducing the need for physical hardware systems. Virtual machines more efficiently use hardware, which lowers the quantities of hardware and associated maintenance costs, and reduces power and cooling demand. They also ease management because virtual hardware does not fail. Administrators can take advantage of virtual environments to simplify backups, disaster recovery, new deployments and basic system administration tasks.

Fault tolerance is concerned with all the techniques necessary to enable a system to tolerate software faults remaining in the system after its development. Fault tolerance is one of the major challenges faced by cloud services for HPC applications. The fault tolerance approach used in high performance and distributed systems is not applicable to cloud because of the resource sharing strategy of cloud.

The practice followed currently for fault tolerance (FT) is checkpoint/restart. However, with increasing error rates, increasing aggregate memory and not proportionally increasing I/O capabilities, this method is becoming less efficient.

The other easiest and cheapest way of dealing with fault tolerance is of dealing with failures is to add some "invulnerable" stations that will oversee the system, assigning work to normal stations and dealing with their failures. The main disadvantage of this solution is, however, creation of a bottleneck. Scalability of such system is bounded by performance of these special stations.

There are number of fault tolerance models which provide different fault tolerance mechanism to enhance the system performance and reliability. There are some drawbacks with the existing fault tolerance models.

Replication is widely used in providing fault tolerance but still suffers from lots of problems such as cost of used resources. The more number of replicas we use, the higher goes the cost of rented resources. More number of replicas may imply number of problems in managing and handling these replicas especially when some of the replica fails.

5. CONCLUSION:

One of The basic mechanisms to achieve the fault tolerance is replication and proactively predicting the faults. Due to replication, cost for renting in cloud resources increase. But it is really required to avoid catastrophic loss. With the number of new replicas increasing, the system maintenance cost will significantly increase, and too many replicas may not increase the availability, but cause unnecessary spending instead. Although proactive techniques suffers lots of drawbacks it is used in most critical system to avoid system crash. Lots of enhancements needed in these areas.

6. REGFERENCES

1. A. Bala and I. Chana, "Fault Tolerance- Challenges, Techniques and Implementation in Cloud Computing", IJCSI International Journal of Computer Science Issues, vol. 9, no. 1, (2012), pp. 1694- 0814.
2. A. Heddaya, A. Helal, "Reliability, Availability, Dependability and Performability: A User-Centered View", Boston, MA, USA, Tech. Rep., 1997
3. Amin, Zeeshan, Harshpreet Singh, and Nisha Sethi. "Review on fault tolerance techniques in cloud computing." International Journal of Computer Applications 116.18 (2015).
4. Avižienis, Algirdas, Jean-Claude Laprie, and Brian Randell. "Dependability and its threats: a taxonomy." Building the Information Society. Springer US, 2004. 91-120.
5. B. Selic, Fault tolerance techniques for distributed systems: http://www.ibm.com/developerworks/ nrational/library/114.html
6. Coenen, J., and Jozef Hooman. "A formal approach to fault-tolerance in distributed real-time systems." Proceedings of the 4th workshop on ACM SIGOPS European workshop. ACM, 1990.
7. F.B. Schneider," Replication Management using the State-Machine Approach," In Sape Mullender , editor, Distributed Systems, pages 169-197, ACM Press, 1993.
8. Ghemawat S, Gobioff H, Leung ST (2003) The Google file system. Oper Syst Rev 37(5):29–43.
9. Gray, Jim, et al. "The dangers of replication and a solution." ACM SIGMOD Record. Vol. 25. No. 2. ACM, 1996.
10. Hosseini, Seyyed Mansur, and Mostafa Ghobaei Arani. "Fault-Tolerance Techniques in Cloud Storage: A Survey." International Journal of Database Theory and Application 8.4 (2015): 183-190.
11. J. Stankovic, "Misconceptions About Real-Time Computing,"IEEE Computer, Vol. 21, No. 10, October 1988, pp. 10-19.
12. Jhawar, Ravi, Vincenzo Piuri, and Marco Santambrogio. "A comprehensive conceptual system-level

approach to fault tolerance in cloud computing." Systems Conference (SysCon), 2012 IEEE International. IEEE, 2012..
13. Jhawar, Ravi, and Vincenzo Piuri. "Fault tolerance and resilience in cloud computing environments." Computer and Information Security Handbook, (2013): 125-141.
14. K. H. Kim, "Towards Integration of Major Design Techniques for Real-Time Fault-Tolerant Computer System",Society for Design & Process Science, USA, 2002.
15. Kaur, Jasbir, and Supriya Kinger. "Analysis of Different Techniques Used For Fault Tolerance." IJCSIT: Int J Comput Technol 4.2 (2013): 737-741.
16. Kim, K. H. "Toward Integration of Major Design Techniques for Real-Time Fault-Tolerant Computer Systems." Journal of Integrated Design & Process Science 6.1 (2002): 83-101.
17. Kim, K. H., and Howard O. Welch. "Distributed execution of recovery blocks: An approach for uniform treatment of hardware and software faults in real-time applications." Computers, IEEE Transactions on 38.5 (1989): 626-636.
18. Kong, Xiangzhen, et al. "Performance, fault-tolerance and scalability analysis of virtual infrastructure management system." Parallel and Distributed Processing with Applications 2009 IEEE International Symposium on. IEEE, 2009.
19. Luckow A, Schnor B. Service replication in grids: ensuring consistency in a dynamic, failure-prone environment. In: Proceedings of IEEE International Symposium on Parallel and Distributed Processing, Miami, 2008. 1–7.
20. Malik, Sheheryar, and Fabrice Huet. "Adaptive fault tolerance in real time cloud computing." Services (SERVICES), 2011 IEEE World Congress on. IEEE, 2011.
21. Malik, Sheheryar, and Jaffar Rehman. "Time Stamped Fault Tolerance in Distributed Real Time Systems." (2005).
22. Merideth M G, Iyengar A, Mikalsen T, et al. Thema: Byzantine fault-tolerant middleware forWeb service applications. In: Proceedings of 24th IEEE Symposium on Reliable Distributed Systems, Orlando, 2005. 131–142.
23. N. Budhiraja, K. Marzullo, F.B. Schneider, and S. Toueg. The Primary-Backup Approach. In SapeMullender, editor, Distributed Systems, pages 199{216. ACM Press, 1993.
24. Nguyên, Toàn, Jean-Antoine Désidéri, and Laurentiu Trifan. "Applications resilience on clouds." High Performance Computing and Simulation (HPCS), 2012 International Conference on. IEEE, 2012.
25. O. S. Unsal, I. Koren, M. Krishna, "Towards Energy-Aware Software Based Fault Tolerance in Real Time Systems"ISLPED '02, Monterey, California, USA, August 12-14,2002.
26. Patra, Prasenjit Kumar, Harshpreet Singh, and Gurpreet Singh. "Fault tolerance techniques and comparative implementation in cloud computing." International Journal of Computer Applications 64.14 (2013).
27. Salatge N, Fabre J-C. Fault tolerance connectors for unreliable Web services. In: Proceedings of 37th International Conference on Dependable Systems and Networks, Edinburgh, 2007. 51–60.
28. Santos G T, Lung L C, Montez C. FTWeb: a fault tolerant infrastructure for Web services. In: Proceedings of 9th IEEE International Conference on Enterprise Computing, Enschede, 2005. 95–105.
29. Sheu G-W, Chang Y-S, Liang D, et al. A fault-tolerant object service on CORBA. In: Proceedings of 17th International Conference on Distributed Computing Systems, Baltimore, 1997. 393.
30. Shvachko K, Kuang H, Radia S, Chansler R (2010) The Hadoop distributed file system. In: Proc 2010 IEEE 26th symposium on mass storage systems and technologies (MSST 2010), May 2010. IEEE Press, New York, pp 1–10
31. Sun, Dawei, et al. "Analyzing, modeling and evaluating dynamic adaptive fault tolerance strategies in cloud computing environments." The Journal of Supercomputing 66.1 (2013):193-228.
32. Tchana, Alain, Laurent Broto, and Daniel Hagimont. "Fault Tolerant Approaches in Cloud Computing Infrastructures." Proceedings of the 8th International Conference on Autonomic and Autonomous Systems (ICAS'12). 2012. (Models Comparison)
33. Thomas Naughton, Engelmann, Christian, Geoffroy R. Vallee and Stephen L. Scott. "Proactive fault tolerance using preemptive migration." In Parallel, Distributed and Network-based Processing, 2009 17th Euromicro International Conference on, pp. 252-257. IEEE, 2009.
34. Tu M, Li P, Yen IL, Thuraisingham BM, Khan L (2010) Secure data objects replication in data grid. IEEE Trans Dependable Secure Comput 7(1):50–64.
35. W. T. Tsai, Q. Shao, X. Sun, J. Elston, "Real Time Service-Oriented Cloud Computing", School of Computing,Informatics and Decision System Engineering Arizona State University USA, http://www.public.asu.edu/~qshao1/doc/RTSOA.pdf.
36. Wang JY, Jea KF (2009) A near-optimal database allocation for reducing the average waiting time in the grid computing environment. Inf Sci 179(21):3772–3790.
37. Wang SS, Yan KQ, Wang SC (2011) Achieving efficient agreement within a dual-failure Cloud computing environment. Expert Syst Appl 38(1):906–915.
38. Yuan D, Yang Y, Liu X, Chen J (2010) A data placement strategy in scientific cloud workflows. Future Gener Comput Syst 26(8):1200–1214.
39. Zheng Z, Zhou TC, Lyu MR, King I (2010) FTCloud: a component ranking framework for Faulttolerant cloud applications. In: Proc 2010 IEEE 21st international symposium on software Reliability engineering (ISSRE 2010), Nov. 2010. IEEE Press, New York, pp 398–407.
40. Zhu Q, Agrawal G, (2010), "Supporting fault-tolerance for time-critical events in distributed environments."Sci Program 18(1):51–76.