IJIRMET ISSN (Online): 2456-0448
International Journal Of Innovative Research In Management, Engineering And Technology
Vol. 1, Issue 5, June 2016

FAULT TOLERANCE IN VARIOUS COMPUTING ENVIRONMENTS

[1] Anchal Mal
[1]Thapar University, Patiala
[1]anchalmal2@gmail.com

Abstract: ***The computational world is becoming very large and complex. There is a blast of new raw data being generated everyday, every hour, every single minute. Today, Google receives 4 million search queries per minute according to the stats given in Data Never Sleeps infographic. Off recent, people have started focusing on reducing computing pro- cessors powers and improve system through- out. Ma jor computing problems have come up in various sectors such as IT and ICT which have lead to the evolution of the pre- viously used, traditional computing environ- ments in order to meet the demands, de- mand for more computational power and storage space. With so much going on, any kind of failure/fault is not acceptable and hence, fault tolerance is the prime need to make computing environments reliable, ro- bust, dependable and available. This pa- per aims at exploring various fault toler- ance methodologies in parallel computing which includes grid, clusters and cloud pro- cessing environments and serial computing which includes homogeneous and heteroge- neous computing environments. Along with this, fault tolerance challenges in ubiquitous computing are also described. This paper is a comparative and intensive study on dis- crete advantages, challenges and issues of fault tolerance in cloud computing. Also, it is an attempt to describe the evolution of the computing frameworks with time.***

Keywords: ***Fault tolerance, Computing en- vironment, Cloud computing, Reliability, Ubiquitous computing, Heterogeneous com- puting, Reactive, Proactive***

I. INTRODUCTION

Computing systems are used in numer- ous applications like defense systems, bank- ing, flight systems, telephone systems etc. Wrong outputs will have different conse- quences leading to inconvenience. Unrelia- bility in any system, computing or otherwise is due to faults in the system. Therefore, fault tolerance is a critical issue in applica- tion and computing platforms. Fault tolerance is a major concern to guar- antee availability and reliability of critical services as well as application execution. When a fault occurs these techniques pro- vide mechanisms to the software to prevent system failure occurrences. To address all such techniques, a comprehensive study has been done. The rest of the paper is organized as fol- lows. Section 2 discusses the nomencla- ture of fault, error and failure along with an overview different aspects of fault toler- ance.Section 3 derives the factors influenc- ing fault tolerance methodologies. Section 4 presents the challenges and issues of fault tolerance in different environments. Section 5 finally concludes the paper.

II. TAXONOMY OF FAULT AND FAULT TOLERANCE

Fault is the inability of a system to do the required job caused by an anomalous state or bug which may be present in one or more than one parts of a system. Faults are the main cause of an error. Different faults are classified as shown in Fig.2

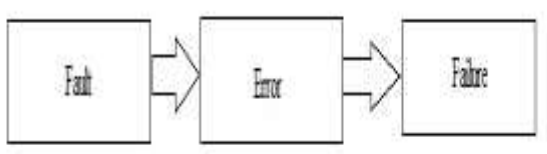

Figure 1: Consequence of a fault

It deals with the art and science of build- ing/working of computing systems that continue performing in presence of faults (one or more in any of the components) satisfactorily. If the operating quality decreases, the decrease is proportional to

IJIRMET ISSN (Online): 2456-0448
International Journal Of Innovative Research In Management, Engineering And Technology
Vol. 1, Issue 5, June 2016

the severity of the fault. A fault tolerant design, enables the system to continue its required task, possibly at reduced level rather than failing com- pletely, when any fault occurs; that is the system doesnt stops completely due to problems either in hardware or soft- ware. For example, a building with a backup electrical generator will provide same voltage to wall outlets even if grid power fails.

III. FACTORS INFLUENCING FAULT TOLERANCE METHODOLOGIES

Effective fault tolerance techniques have many metrics in its account as follows:

• Scalability: It determines the capabil- ity of an algorithm to tolerate the faults with given number of nodes.

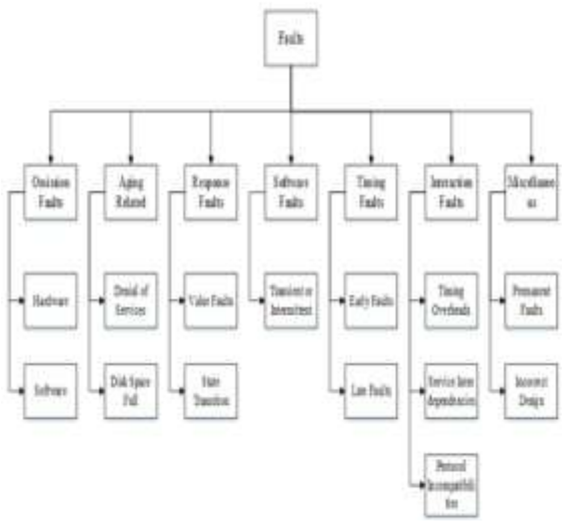

Figure 2: Nomenclature of faults

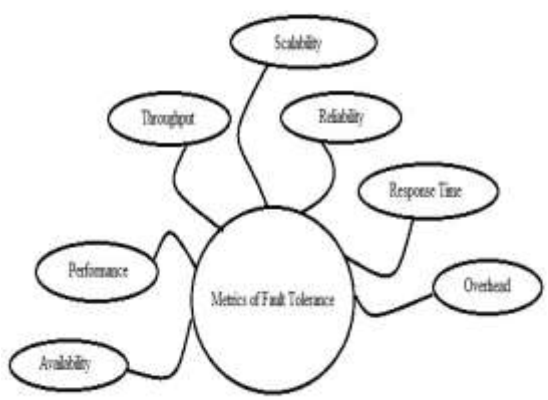

Figure 3: Metrics for fault tolerance

Table 1: Table showing two ma jor types of fault tolerance [1]

Hardware fault tolerance	Software fault tolerance
Most of fault-tolerant strategies have fo- cused towards structuring systems that can recover themselves from the faults that usually occur in hardware modules, this involves splitting a computing sys- tem into modules. So if a particular module gets failed, other module can continue its functioning.	It is similar to hardware approach but here more consideration is on tolerating faults at the software level. For achiev- ing this various static and dynamic re- dundancy approaches are used.

• Reliability: It aims to give accurate out- puts within specified period of time.

• Response time: It is the time taken by a particular algorithm to respond.

• Throughput: It is the number of tasks which have completed their execution.

• Performance: It is used to check profi- ciency of the system.

• Overhead: It describes the amount of overhead involved while executing a fault tolerance algorithm.

• Availability: The possibility that a par- ticular job will function adequately at a given amount of time can be considered in terms of availability of resources.

IV. CHALLENGES IN VARIOUS FAULT TOLERANCE TECHNIQUES

• Homogeneous Computing

Computing systems have evolved at a fast pace, homogeneous serial systems [2] were first used which guaranteed sim- ilar results and storage on each hard- ware processor, same results for floating point numbers; even the software (op- erating system, compiler, compiler op- tions) on each processor also guaran- tees the same storage representation and the same same results for operations on floating point numbers.

Hardware, time, information, and soft- ware redundancy are used for fault- tolerance. Of the many structures used in hardware fault-tolerance [3], two stand out.

• Heterogeneous Computing

It refers to systems that use more than one kind of processors and cores. These computational units (general purpose processor, special purpose processor or a co processor) make the systems per- form better. Heterogeneity [4] here was basically in context of different instruc- tion set architectures, where main pro- cessor had one and rest had different, consumed high power. The addition of the extra computational units makes this system similar as parallel comput- ing or multi-core computing systems and hence, more tasks are being completed per unit time. Real time fault tolerant scheduling algorithms [5] are used.

• Grid Computing

It offers sharing of resources over geo- graphically distributed locations, a com- puter network in which each computer's resources are shared with every other computer in the system. Moreover, col- laborative nature of grids leads to con- cept of virtual organizations(VO) which work towards a particular task. Various fault tolerance techniques at application levels have been proposed.

IJIRMET ISSN (Online): 2456-0448
International Journal Of Innovative Research In Management, Engineering And Technology
Vol. 1, Issue 5, June 2016

• Cloud Computing

Cloud Computing is a concept which refers to services and applications which are executed on a distributed network with the help of resources which are vir- tualized. Cloud refers to somewhere up there, with huge amount of flexible re- sources that can be used whenever re- quired.

Table 2: Table showing fault tolerance techniques in heterogeneous environment

System level Checkpoint/ Message Log- ging [6]	The idea is to incorporate fault toler- ance in the system level so that applica- tion can be recovered automatically.
Compiler based fault tolerance	It is a transparent approach in which compiler inserts the checkpoint at the best place and to exclude irrelevant memory to reduce the size of check- point.
User Level checkpoint libraries	The idea is to provide some checkpoint libraries programmer and let the pro- grammer decide where, what to check- point.
Algorithmic fault tolerance approach	The idea is to exploit the knowledge of algorithms to reduce fault tolerance overhead to the minimum.

Figure 4: Pervasive Computing

Two broad classifications of fault tol- erance techniques in cloud: 1. Reac- tive [7]: It reduces the consequence of failures on application execution when the failure effectually occurs. 2. Proac- tive [8]: Principle of this method is to avoid recovery from faults, errors and failures by predicting them and replac- ing the doubted components with other working components.

• Ubiquitous Computing

Currently, pervasive computing [22] is trending. Pervasive computing goes be- yond the realm of personal computers: it is the idea that almost any device, can be embedded with chips to connect the device to network of other devices. Since pervasive computing exists in the user's environment, the technology is sustain- able if it is invisible to the user and does

IJIRMET ISSN (Online): 2456-0448
International Journal Of Innovative Research In Management, Engineering And Technology
Vol. 1, Issue 5, June 2016

not intrude the user's conscious- ness. A pervasive system consists of different kinds of devices such as desk- tops, laptops, handhelds, sensors, actu- ators, displays, speakers, scanners, cam- eras and pro jectors etc.

Therefore, the system needs to be re- silient to various faults, an application or device that stops on failure can be de- tected through timeout techniques such as heartbeat messages. Once a fault is identified, it should be isolated to pre- vent its propagation to other parts of the system, faults should be tolerated with minimal user awareness.

V. CONCLUSION

Tolerance of faults makes an important problem in the scope of environments of computing. In the present scene, there are number of models which provide different mechanisms to improve the system and pro- vide reliability; only the most efficient ones have been discussed in this paper. But still there are number of problems which re- quires some concern for every frame work. Computing systems have evolved at a fast pace and now the entire focus has shifted

Table 3: Table showing cloud computing fault tolerance techniques

Fault toler- ance tech- niques	Category	Major features	Tools Used	Types of faults detected
Check pointing/ Restart [9]	Reactive	When a task fails, it is al- lowed to be restarted from the recently checked pointed state rather than from the beginning. It is an efficient task level fault tolerance.	SHelp [10]	Application fail- ure
Replication [11]	Reactive	Various task replicas are run on different resources, for the execution to succeed till the entire replicated task is not crashed.	HA-Proxy [12], Hadoop [13], AmazonEc2	Node fail- ure,Process failure
Job Migra- tion [14]	Reactive	During failure of any task, it can be migrated to another machine.	HA-Proxy	Node fail- ure,Process failure
S-guard [15]	Reactive	It is based on rollback re- covery less disruptive to nor- mal stream processing and makes more resources avail- able.	Hadoop	Application fail- ure,Node failure
Retry [16]	Reactive	It retries the failed task on the same cloud resource.	Assure [17]	Netwok fail- ure,Host failure
Task re- submission [18]	Reactive	Whenever a failed task is de- tected, it is resubmitted ei- ther to the same or to a dif- ferent resource at runtime.	AmazonEc2	Node fail- ure,Application failure
Rescue work- flow [19]	Reactive	It is a technique in which workflow to continue even if the task fails until it be- comes impossible to move forward without catering the failed task.	Hadoop	Node fail- ure,Application failure
Self healing [20]	Proactive	When multiple instances of an application are running on multiple virtual ma- chines, it automatically han- dles failure of application in- stances.	Assue	Netwok fail- ure,Host failure
Preemptive migra- tion [21]	Proactive	It relies on a feedback-loop control mechanism where application is constantly monitored and analyzed.	HA-Proxy	Node fail- ure,Process failure

Table 4: Table showing fault tolerance techniques in pervasive environment

Device failures	Each device has its own set of faults that can potentially contribute to the fail- ure of the pervasive system. Mobile de- vices have physical constraints such as finite battery power and limited signal strength. So if the battery goes down or if the signal strength is too low they get disconnected from the pervasive system and are regarded as having failed.
Application failures	Even in well-tested software systems, bugs of varying severity are found . Per- vasive computing includes commercial off-the-shelf applications that may not be well tested. In some situations, ap- plications may work well as stand-alone software but may not inter-operate cor- rectly or reliably with other software.
Network failures	Pervasive systems consist of wired and wireless devices. Therefore, a reli- able pervasive system should account for network failures caused by low signal strength, devices going out of range and unavailability of communication chan- nels due to heavy traffic. Network fail- ures lead to unreachable devices that may be wrongly perceived as device fail- ures.

to cloud and pervasive computing. This pa- per discussed the fault tolerance techniques covering its research challenges, tools used for implementing fault tolerance techniques in computing environments along with how computing systems have evolved over time.

References

[1] B. Randell, “System structure for software fault tolerance,” in ACM SIGPLAN Notices, vol. 10, no. 6. ACM, 1975, pp. 437–449.

[2] M. Taufer, D. Anderson, P. Cicotti, and C. L. Brooks III, “Homogeneous redundancy: a technique to ensure integrity of molecular simulation results using public computing,” in Parallel and Distributed Processing Sympo- sium, 2005. Proceedings. 19th IEEE Interna- tional. IEEE, 2005, pp. 119a–119a.

[3] A. Avižienis and J. P. Kelly, “Fault toler- ance by design diversity: Concepts and exper- iments,” Computer, vol. 17, no. 8, pp. 67–80,1984.

[4] A. A. Khokhar, V. K. Prasanna, M. E. Shaa- ban, and C.-L. Wang, “Heterogeneous com- puting: Challenges and opportunities,” Com- puter, no. 6, pp. 18–27, 1993.

[5] H. Topcuoglu, S. Hariri, and M.-y. Wu, “Performance-effective and low-complexity task scheduling for heterogeneous comput- ing,” Parallel and Distributed Systems, IEEE Transactions on, vol. 13, no. 3, pp. 260–274, 2002.

[6] S. Yi, A. Andrzejak, and D. Kondo, “Mone- tary cost-aware checkpointing and migration on amazon cloud spot instances,” Services Computing, IEEE Transactions on, vol. 5, no. 4, pp. 512–524, 2012.

[7] S. Sidiroglou, M. E. Locasto, S. W. Boyd, and A. D. Keromytis, “Building a reactive immune system for software services,” in Proceedings of the general track, 2005 USENIX annual tech- nical conference: April 10-15, 2005, Anaheim, CA, USA. USENIX, 2005, pp. 149–161.

[8] G. Vallee, C. Engelmann, A. Tikotekar, T. Naughton, K. Charoenpornwattana, C. Leangsuksun, and S. L. Scott, "A frame- work for proactive fault tolerance," in Availability, Reliability and Security, 2008. ARES 08. Third International Conference on. IEEE, 2008, pp. 659–664.

[9] S. K. Mondal, F. Machida, and J. K. Muppala, "Service reliability enhancement in cloud by checkpointing and replication," in Principles of Performance and Reliability Modeling and Evaluation. Springer, 2016, pp. 425–448.

[10] G. Chen, H. Jin, D. Zou, B. B. Zhou, and W. Qiang, "A lightweight software fault- tolerance system in the cloud environment," Concurrency and Computation: Practice and Experience, vol. 27, no. 12, pp. 2982–2998, 2015.

[11] H. Goudarzi and M. Pedram, "Energy-efficient virtual machine replication and placement in a cloud computing system," in Cloud Computing (CLOUD), 2012 IEEE 5th International Con- ference on. IEEE, 2012, pp. 750–757.

[12] P. K. Patra, H. Singh, and G. Singh, "Fault tolerance techniques and comparative imple- mentation in cloud computing," International Journal of Computer Applications, vol. 64, no. 14, 2013.

[13] K. Kambatla, A. Pathak, and H. Pucha, "To- wards optimizing hadoop provisioning in the cloud." HotCloud, vol. 9, p. 12, 2009.

[14] I. Brandic, "Towards self-manageable cloud services," in Computer Software and Applica- tions Conference, 2009. COMPSAC'09. 33rd Annual IEEE International, vol. 2. IEEE, 2009, pp. 128–133.

[15] P. K. Patra, H. Singh, and G. Singh, "Fault tolerance techniques and comparative imple- mentation in cloud computing," International Journal of Computer Applications, vol. 64, no. 14, 2013.

[16] M. Abu Sharkh, M. Jammal, A. Shami, and A. Ouda, "Resource allocation in a network- based cloud computing environment: de- sign challenges," Communications Magazine, IEEE, vol. 51, no. 11, pp. 46–52, 2013.

[17] B. P. Rimal, E. Choi, and I. Lumb, "A taxon- omy and survey of cloud computing systems," in 2009 Fifth International Joint Conference on INC, IMS and IDC. Ieee, 2009, pp. 44–51.

[18] K. Plankensteiner, R. Prodan, and T. Fahringer, "A new fault tolerance heuristic for scientific workflows in highly distributed environments based on resubmission impact," in 2009 Fifth IEEE International Conference on e-Science. IEEE, 2009, pp. 313–320.

[19] E. Sindrilaru, A. Costan, and V. Cristea, "Fault tolerance and recovery in grid workflow management systems," in Complex, Intelligent and Software Intensive Systems (CISIS), 2010 International Conference on. IEEE, 2010, pp. 475–480.

[20] Y. Dai, Y. Xiang, and G. Zhang, "Self-healing and hybrid diagnosis in cloud computing," in Cloud computing. Springer, 2009, pp. 45–56.

[21] R. Santhosh and T. Ravichandran, "Pre- emptive scheduling of on-line real time ser- vices with task migration for cloud comput- ing," in Pattern Recognition, Informatics and Mobile Engineering (PRIME), 2013 Interna- tional Conference on. IEEE, 2013, pp. 271–276.

[22] D. Saha and A. Mukherjee, "Pervasive com- puting: a paradigm for the 21st century," Computer, vol. 36, no. 3, pp. 25–31, 2003.