Analysis of reactive and proactive fault tolerance mechanism in cloud computing

Neha [1], Kamaljit Kaur [2]
Computer Engineering and Technology,
Guru Nanak Dev University, Amritsar, India

Abstract :- Cloud computing demand is expanding because of which it is essential to give redress benefits within the sight of faults too. The Resources in cloud computing can be progressively scaled that too in a manner that is cost effective. Adaptation to non-critical failure is the way toward discovering faults and failures in a system. The system should work appropriately even if there is chance of failure happens or hardware failure or programming failure. Failure ought to be overseen viably for dependable Cloud Computing. It will likewise guarantee accessibility and robustness. This paper expects to give a superior comprehension of adaptation to non-critical failure procedures which are utilized for overse*eing faults in cloud. It additionally manages some current Fault tolerance model.

Keywords*:* Fault Tolerance,Cloud computing,Reactive,Proactive

Introduction

In cloud computing the assets are gotten to remotely here and there which makes faults amid that procedure. To amend the defective segments adaptation to non-critical failure techniques are utilized and adjust them. Because of its remote access there are bunches of odds of mistakes so to accomplish unwavering quality progressively cloud computing. Adaptation to internal failure is accomplished by mistake handling having two constituent stages. These stages are "viable mistake forms" i.e. prior to the event of blunder and "dormant mistake handling" implies mistake won't happen once more. Cloud computing describes the abstraction of web based computers, resources ,and services. It basically allocates the computing resources over the internet. Cloud computing provides computing services to its users such as servers, storage, database, networks, software and more over the internet which can also be termed as 'the cloud'. It brings evolution to grid based computing. The benefits of cloud computing are on-demand services, scalability and reliability[1]. Cloud computing can be afford by users as it is provided in a Pay-As-You-Use manner to users. Cloud is essentially provided by large distributed data centers. As the people are becoming more dependable on their gadgets for services, data storage has become everyone priority. The main reason for adopting cloud computing by many organizations is cost effectiveness. Cloud computing has make it easier for large and small scale for storing data at low cost. Cloud computing removes the need for task like hardware set up so that user and service provider can spend more time on achieving important business goals. Users who subscribe to computing services delivered via the cloud are able to gain access to fast and flexible enterprise level computing services. Users interaction with the cloud through their web browser, eliminates the need for installing numerous software applications on hardware which eventually solves the storage related issues. It not only simplify in house computing operations, but also change the way products are deployed to its users.The cloud delivers storage and applications as a service but not a product.

Service models of cloud computing

The various service models that cloud providers offered are mainly categorized in three categories:

1. **Software as a Service (SaaS)**

This model provides complete application to the user on demand. It can handle multiple users at a time but in background only single application of the service is executed. The single application facilitate the users, they need not to go for any advance payment. Various service provider like Google, salesforce , Microsoft provides Saas facility.

2. **Platform as a Service(Paas)**

This model provides environment as a service to user for constructing own application that run on service provider base. It also provides predefined application server combined with OS to user. Google's App Engine, Force.com is providing a platform to users.

3. **Infrastructure as a Services(IaaS)**

This model provides services to fundamental storage and computing capability. In order to manage workload the shared resources are utilized among various users. To use the services the user has install own software over the infrastructure. Amazon and GoGrid are some example of IaaS.

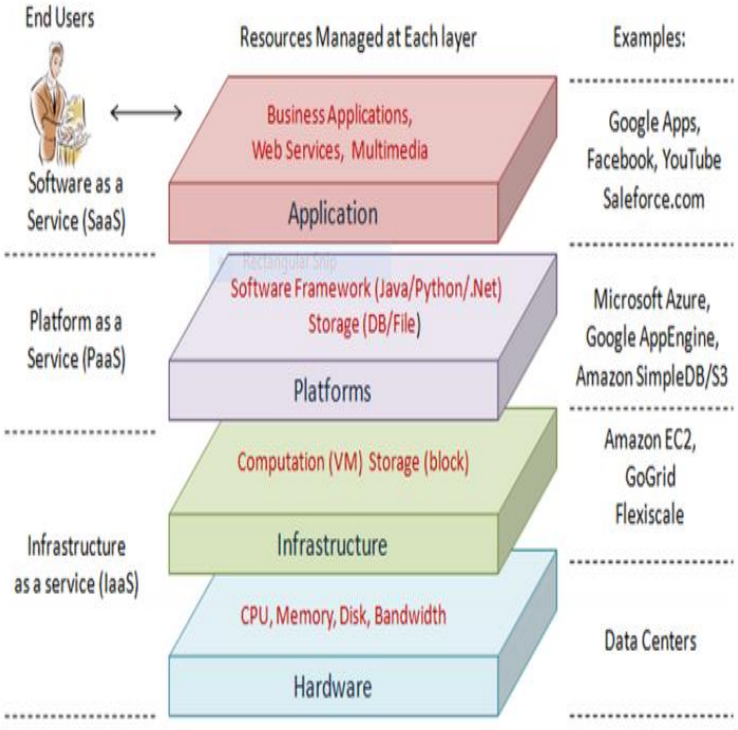

Fig. services of clouds[2]

Quality of Service (QOS) plays a vital role in cloud computing . According to Quality of Service reliability and performance are necessary features to be provided in cloud computing paradigm. As the customers access the services remotely and data is located on data centres through internet instead locally the risk of errors and failures increases. The service provided to customers should achieve reliability and better performance. These parameters can be achieved through fault tolerance.[3] Fault tolerance is the ability of system to perform functionality even in the presence of failures. Fault tolerance is major concern for both customers and service customers. A fault tolerant system should be capable of handling faults in individual hardware or software or power failures or any kind of bizarre behaviour and still continue its processing. The basic problem is as the complications of system increases, its reliability decreases if necessary measures are not taken. But even in the case when the system is designed and

implemented perfectly there can be possibility of faults. Fault tolerance have been noticed for many years, there are different techniques of fault tolerance which can be implied according to the user requirements and their applications. The main idea of using fault tolerance is to focus on masking of faults instead of completely removing or fixing them[4]. Fault tolerant should be used for defaulted components and in spite of implementing fault tolerant mechanism in all components as it will reduce cost of installation. A system capable of fault tolerance should attain parameters including reliability, availability and performance . Reliability and performance are important as low reliability and low performance can cause failures and become a cause of disappointment to customers. By providing high reliability and high availability leads to robust fault tolerance during development time. High reliability is quite difficult to achieve as there are large number of physical servers and virtual machines in cloud whose risk of failure is high. The focus on availability and reliability concern can improve system safety[5]. The main contribution of this paper is to give a general survey of fault tolerance techniques.Section II gives an overview of fault tolerance and discuss types of faults and failure and section III gives literature review of earlier work done on fault tolerance whereas section IV identifies general parameters involved in fault tolerance and in section IV have compared parameters achieved in techniques by the research expertise.

Fault can cause system unable to do its required task caused due to presence of bug in one or more that one part of system. The main cause behind the occurrence of error is fault. Fault can be a physical defect or flaw in software component or hardware component example a software bug .Faults can cause errors and errors can further leads to cause failures. Faults can be classified on various factors such as network ,processor, process,media and physical faults. Faults can be caused due to presence of software bug or external behaviour or hardware faults.

- ✓ ***Physical faults*:** The Fault which takes place in hardware like fault in CPUs, Fault in memory, Fault in disk,virtual machine etc.
- ✓ ***Intentional faults:*** faults that are caused to system on purpose by inserting malicious code.
- ✓ ***Internal faults:*** faults which happen due to presence of software bug or internal design.
- ✓ **External faults:** faults which can cause serious outcomes due to some external behaviour.
- ✓ ***Design faults* :** manufacturing defects in caused by fuzzy human behaviour can cause various failures.
- ✓ ***Operational faults:*** are also caused in phase of creation which can cause operational fault like application failure or Vms failure.
- ✓ ***Software faults:*** occur due to presence of bug or design fault or due to fuzzy human factors which are harder to prevent.
- ✓ ***Permanent faults:*** are the faults that once occurred then continues to exist until replaced or repair. They are caused due to Power failure and they can cause major disruptions.Permanent ,Intermittent and transient comes under the category of timing faults.
- ✓ ***Intermittent fault:*** are the one that occurs occasionally, they sometime vanish and then reappears so it become quite difficult to detect them.
- ✓ ***Transient faults:*** are the faults caused by some inherent fault in system, if operation is repeated several times then fault goes away and corrected by rollback to previous state or by restarting the system. Faults are also classified on their behaviour into Byzantine faults and Crash faults.
- ✓ ***Byzantine faults:*** are the faults which cause system to behave arbitrarily at arbitrary times such that a system do no stop execution even after failure but gives wrong output.
- ✓ ***Crash faults:*** are the faults which once occurred make the system halt completely[6].

A. ERROR

The presence of fault can cause the system component to behave erroneously. Error can be defined as predicted difference between its observed value and actual value . Faults can cause errors and errors can further leads to cause failures. Errors are

related with incorrect values in the system state . There are different types of errors that can be caused like network errors ,software errors and miscellaneous errors.

B. FAILURES

The misconduct of system when system reaches an invalid statement observed by human or by system itself. Failures takes place due to an error which leads to reaching an invalid state. There are many classification of failures that have been developed. Failure different types includes crash failure,omission failure ,timing failure ,omission failure ,response failure and many other following discussed .

1. *Crash failure* are the failures in which system halts but work accurately until it stopped working .Once the system crashes, it do not perform any further results..It comes under the category of permanent omission failure. For example an operating system comes to halt situation and the only solution left is to reboot it.

2. *Omission failure* when the server lacks in getting responses or fails to respond to a request for example not getting response or failed messages due to overloaded server and When the server is not getting enough responses then the current server is not affected directly as the server is not known about the messages are not received yet. Some other types of omission faults in software can make the system hang caused by infinite loops .

3. *Timing failures* is one of classification of which occurs when response do not reach the server at the specified time interval that it intended to reach which include delay faults and early faults.

4. *Response failures* are the one in which the response received is incorrect completely for example a search engine that do return web pages not related to searched data or incorrect web pages are returned [7].

5. *Arbitrary failures* : The most serious failures are Byzantine failures or arbitrary failures failures which deviates from intended processing according to an algorithm due to bug or malicious attack.In particular, it may happen that a server is producing output it should never have but which cannot be detected as being incorrect.It is quite clear that byzantine or arbitrary failures are the worst kind of faults. Failures that cause halting of server can be classified into different types.

6. *Fail-stop failures* are the one of crash failure which makes the server halt and can be easily detected and results in permanent failures.

7. *Fail silent failures* are the failures in which the process cannot differentiate between crash failures and omission failure.

8. *Fail-safe failures* deal with arbitrary failures and do not cause any kind of harm. It is quite clear that byzantine or arbitrary failures are the worst kind of faults.

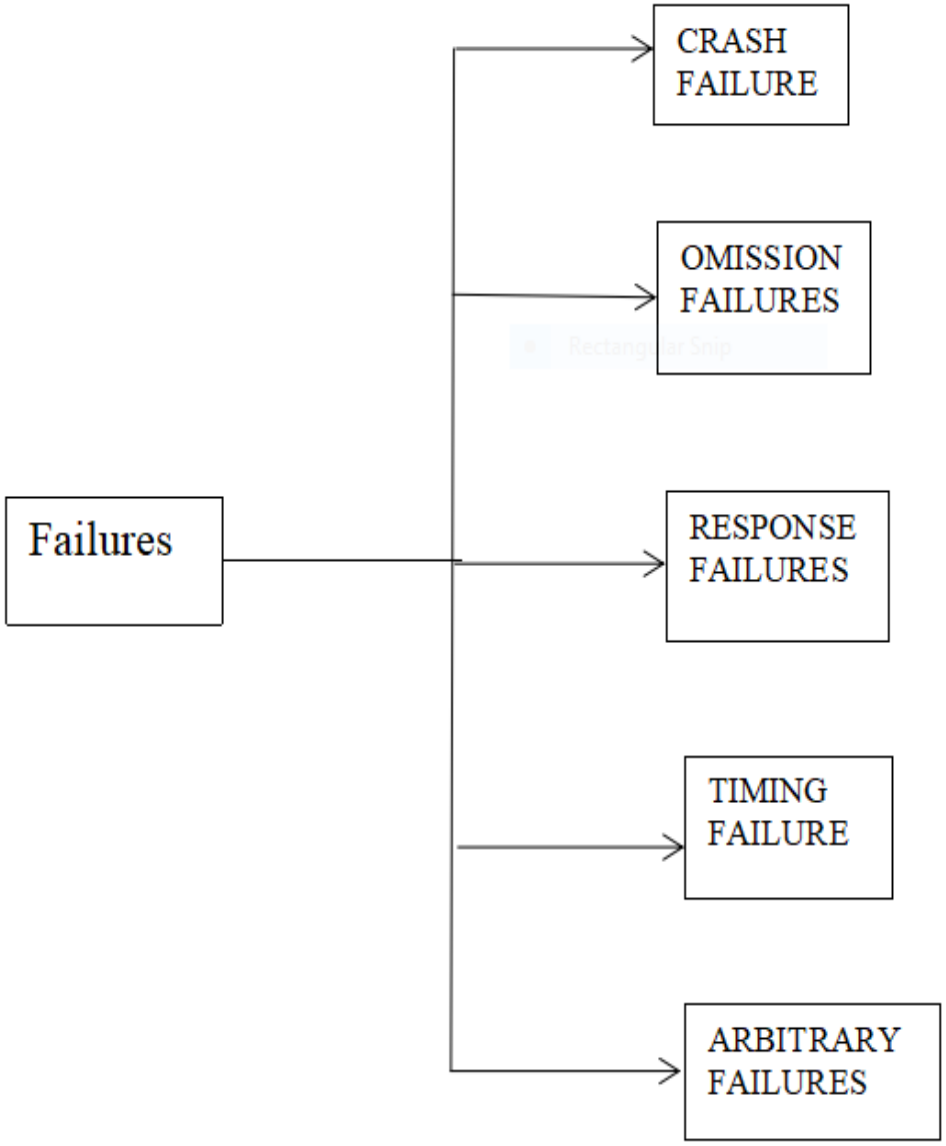

Fig. Types of failures under mixed category

9. *Fail silent failures* are the failures in which the process cannot differentiate between crash failures and omission failure.

10. *Fail-safe failures* deal with arbitrary failures and do not cause any kind of harm. It is quite clear that byzantine or arbitrary failures are the worst kind of faults

FAILURE MEASURE:

There are **common measures** of failures includes: Failure Rate , Mean time to Failure, Mean time to Repair and Mean Time between Failure and defined as follows:

Failure rate can be defined as expected number of failures in per unit time and it changes with time proportionally.

Fault Tolerance : fault tolerance is the ability of the system continue its execution even in the presence of failures and to avoid service failures by reducing their impact on the cloud. A system is said to fail when it do not perform the way it is intended to.[3]

Diverse adaptation to non-critical failure methods and systems, we order faults as takes after:

- Proactive adaptation to internal failure: The Proactive adaptation to non-critical failure intends to recognize the issue before it really happens. In this instrument, foresee the blame and maintain a strategic distance from recuperation from

blame, blunders and failure. It keeps the parallel running applications from getting influenced because of failure of influenced single segments.

- Reactive adaptation to non-critical failure: Reactive adaptation to non-critical failure system decreases the exertion of failures, when failure happens. This On-request adaptation to non-critical failure strategy makes framework more robust.
- Adaptive adaptation to internal failure: The versatile adaptation to non-critical failure methods are done naturally as indicated by the circumstance. Adaptive Fault Tolerance (AFT) can guarantee unwavering quality even under basic circumstances.

***Different fault tolerance techniques* :**

There are different techniques that helps in handling faults which comes under Reactive , Proactive and Adaptive fault tolerance techniques which are discussed further below .Many authors have described different techniques of fault tolerance and parameters achieved with a particular fault tolerance technique including reliability and performance

1. Reactive fault tolerance

The impact of failures can be reduced on the system that leads to cost and effort effectiveness which can be achieved through reactive fault tolerance. Reactive fault tolerance is used after fault has occurred.It helps in improving from unstable to stable state so that expected results can be achieved . It helps in providing system which means the system can handle variability and remain effective. It includes checkpointing , replication, self healing which are defined below.

1) *Checkpointing:* It is a technique in which whenever a system encounters a failure ,it is restored to previously correct state using latest checkpoint. The technique work on basis

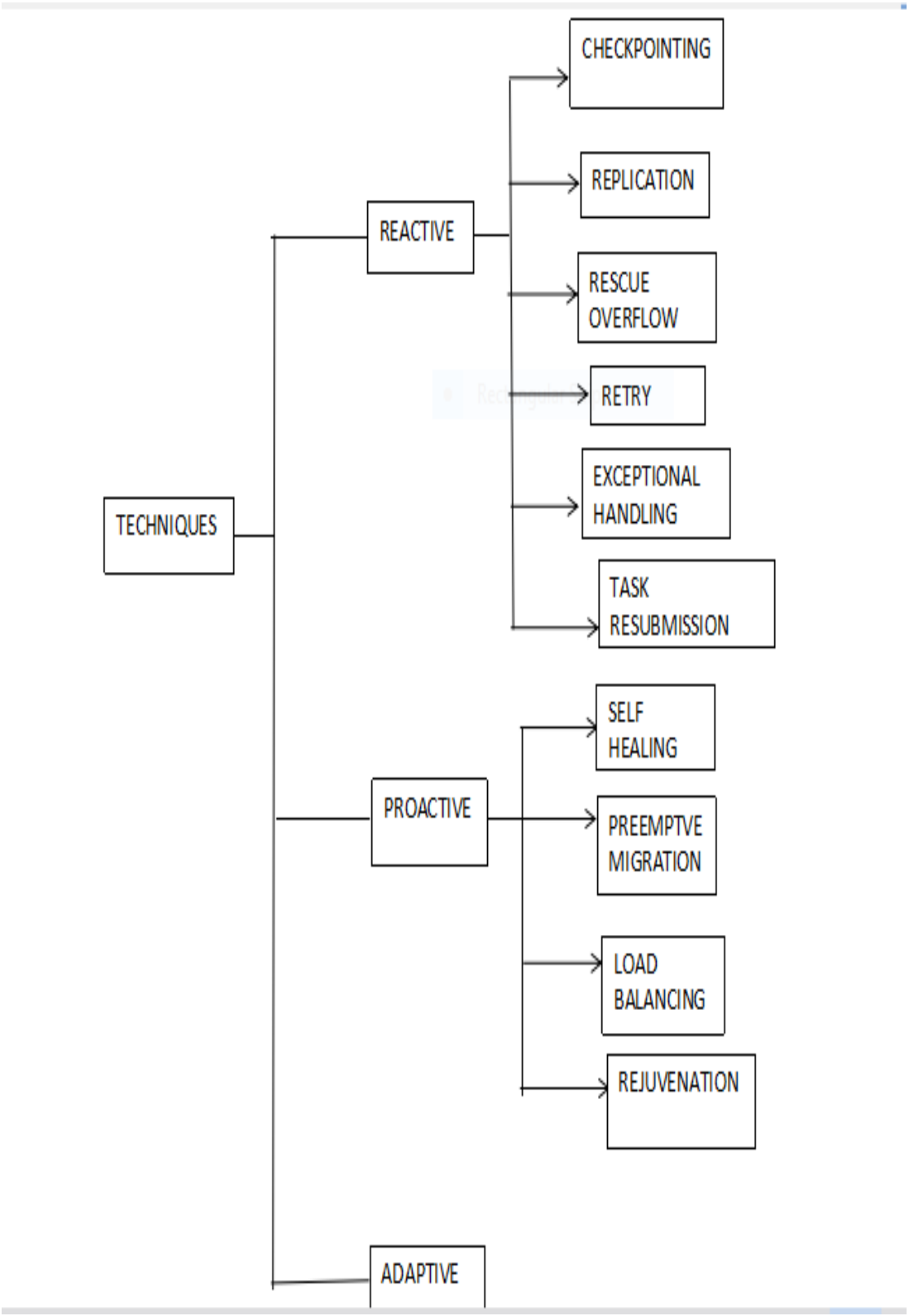

Fig. Techniques of fault tolerance

of saved state by system at different time intervals in cache memory and when system undergoes a failure while the execution it can be recovered from last saved state rather than restarting or rebooting the system .But most of the time ,saving state of the system at different intervals can cause execution time overhead which can be considered as its demerit. So checkpoint overhead should be minimized.Checkpoints can be for local (for separate process) or global (for set of processes).

2) *Replication:* Replication is a technique in which job or data is copied in different machines such that if one fails then other copies can be perform the execution in similar manner. This process confirms consistency between redundant resources Replication can be further classified into passive and active replication. *Passive* replication is one in which only one primary replica is there which performs execution while the others are backup replicas which save the state and only take over the job if primary fails. *Active* replication can leads to simultaneously invoking of all replicas and process same request. Replication example tools includes Zerto which enables replication of hypervisor across range of machines. Replication helps to succeed in achieving fault tolerance , reliability and accessibility

3) *Rescue workflow*: this technique helps the workflow to not to stop and continue execution even if the tasks fails until it become impossible to move forward. It allows processes to continue to execute even in the presence of failure .[11]

4) *Self healing* : this technique enables the cloud machines capable of detecting errors and failures and localizing the components which are faulty one and healing those components with recovering algorithm.Multiple instance of application are installed and run on different virtual machines and if failure occurs it can be controlled automatically through different machines.

5) *Retry:* this technique is the simplest one in which task is resubmitted by the user with the request of processing again to the cloud.

6) *Exceptional handling* :In this method the user decided how to handle failures in the system and what steps to need to be taken.

7) *Task Re-submission*: during failure occurrence, the task is submitted to same or different machine.[9]

2. **Proactive fault tolerance**

The proactive fault tolerance helps to avoid faults before the occurrence of faults. In this technique the occurrence of faults are predicted before they takes place and replace the faulty components with non faulty components. It avoids recovery from faults. It includes preemptive migration, Load balancing and software rejuvenation discussed briefly below .

1) *Preemptive migration:* whenever any failure occurs of any task , it can be migrated to another machine which can be implemented using tools like HAProxy, Hadoop.[

2) *Rejuvenation :* It is quite similar to reboot of the system and the rebooting is done after periodic intervals. Here virtual machines running in cloud environment are restarted and operating system is reloaded with clean state .[10]

Load balancing : whenever the load of CPU and memory exceeds a certain limit for example if 75% utilization of CPU is the limit and if it exceeds then the load of CPU with exceeded limit is transferred to other CPU

Literature Survey

[1]covers the fault tolerance techniques in cloud computing along with its challenges and implementation. It also proposes HAproxy based cloud virtualized system. This system deals with faults that occur in system along with server application in cloud. If there occurs fault in any server over a connection then it automatically redirects to another one. Also data replication technique utilized.

[2]Paper proposes technique which is reliability based, for fault tolerance. The technique is AFTRC model that keep track of all nodes reliability. This technique deletes the node that has less reliability and the checkpoints are used. If any fault occurs during the transformation of data then backward recovery has been done by these checkpoints.

[3]Proposes availability based fault tolerance technique that uses FTM model. This technique uses replication of users and when replica fails it use the algorithm like gossip based protocol.

[4] This paper gives two visions for management of fault in cloud computing platforms. In the first one it provide management over the cloud provider and second one will share responsibilities over two participants over cloud. This paper has result that shows the improvement over the cloud. Also involve exclusive and collaborative fault tolerance techniques.

[5] Describes the study of different fault tolerant technique and describes different types of failure. It describes how fault can be occurs in system and which action should be taken if fault could occur.

[6] It proposes FTC technique for fault tolerance which uses check pointing. This is good mechanism for fault that occurred in cloud computing over the infrastructure. The advantage of method is forward recovery that is provided with the help of checkpoints. The results show that this scheme has given reliability and less chances of failure.

[7]studies models for fault tolerance and also compare them on the basis of metrics for fault tolerance. There are many techniques for fault tolerance in cloud computing and these techniques use different criteria's for finding faults. But still there are challenges that need some concern to speed up fault tolerance.

[8] describes technique that is based on scheduling using CAN in mobile computing. The CAN (content address networking) logically manages the locations of various devices the uses clouds for storing their data. Also malicious user filtering is done that ensure no unauthorized user could access the cloud.

[9]describes the system that overcome the problem of energy efficiency and fault tolerance. Each fragment is assigned to node so that energy consumption is less during data retrieval. The result shows that implementation of the system gives feasible solutions.

[10]presented four crucial abstractions in the space of fault tolerant disseminated frameworks. Message Abstractions address the rightness of individual messages sent and got. Fault Abstractions address the sorts of faults conceivable and also their belongings in the framework. Fault-Masking Abstractions address the sorts of nearby calculations forms make to cover faults. At long last, Communication Abstractions address the sorts of information conveyed and the properties required for correspondence to prevail within the sight of faults

[11]suggests that the capacity of fault tolerance is to safeguard the conveyance of anticipated administrations in spite of the nearness of fault-caused blunders inside the framework itself. Mistakes are identified and amended, and lasting faults are found and expelled while the framework keeps on conveying worthy administration. This objective is expert by the utilization of mistake recognition calculations, fault analysis, recuperation calculations and extra assets.

Technology advancement led to situation in which machines having least resources can execute more using data centers in cloud computing. Pay per use services are provided by cloud computing [12]and advanced computing mechanisms. As dependability increase so does a risk. This risk arises as more and more nodes starts to interact with advance computing model. Mean time between failure increases as more and more nodes interact with cloud and advance computing. Hence trade off exist between number of nodes and reliability. As process fail all the progress made by the node is lost. fault tolerant strategy is required to cope with this situation. Re-execution of process can lead to recovery and hence lost progress can be recovered. Mechanism following the listed approach is known as Time dependent mechanism[13][14]. Strategies under Time redundant approach are described in this section

- Computing with migration

This is one of commonly used mechanism to ensure fault tolerance. Functions and operands are recomputed and compared against the previous results to ensure accuracy and faults present within the system. This mechanism is effective enough for short term faults that existed within the system. This approach however fails if faults is permanent or long lasting within the system[15]. For

example, consider addition of two numbers X_1 and X_2. Numbers are added by shifting the digits of numbers to the left and then added again. This process continues until corrected answer is not obtained. Mechanism is reliable in case of short term or temporary faults.

- **CHECKPOINTS**

This is popular mechanism of establishing fault tolerance within advance computing schemes. In this approach save point is established at various points in time. As the node is using the resource reaches certain point in time where save point is established then progress is saved. This is a automated approach and has many advantages. One of the advantages is fault tolerant ability is introduced through recovery approach. The drawback of this approach is that it will consume more time to recover a system to normal stage[16][17].

- **STATIC FAULT TOLERANCE**

Passive fault tolerance do not necessarily remove fault from the system rather it hide the fault and hence fault become invisible. Mechanism is also termed as fault masking. Major problem with the passive fault tolerance is small problem in inputs can yield large deviation in output. Voting mechanism is used in order to select resources in this approach. The voting could either be hardware or software in nature. Hardware voting is fast but expensive. Software voting is flexible but slow[18][19].Mid value select approach is used in order to rectify voting problem.

- **DYNAMIC FAULT TOLERANCE**

Active fault tolerance mechanisms are actually used to rectify the faults that occurred within the system. Hence faults are actually removed rather than hiding as in passive fault tolerance. Active fault tolerance mechanism involve the following techniques

a. REPLICATION

In this approach two identical modules performing the same task are placed in parallel. The results produced by two identical modules are compared with each other. The threshold value is established. The module producing result satisfying threshold is accepted. The problem with this approach is that it can only detect the faults but cannot tolerate. Recovery mechanism is hence absent in this case[20][19].

b. REJUVINATION

This approach utilizes spare modules along with workable module. One module in this case is operational while other modules are spare. Error detection mechanisms are implemented to detect when the fault is occurring within the system along with detection of module in which fault occurred. The faulty module is removed from the system and is replaced by the spare module. Special switch is used to monitor the errors. The module if error frees then selection is made on the basis of priority. The module having error is eliminated from the system[21][10].

Hot standby sparing is also available in which spare modules are active all the time and are ready to change place with running module in case of errors. This switching operation is fast and downtime is close to zero. This downtime is the problem with offline standby sparing techniques[10].

c. EXAMINE THE OPTIMAL MACHINE FOR MIGRATION

In this technique hybrid approach is followed by combining duplication with comparison and stand by sparing. Two modules are operated in parallel rather than one module. Their results are compared and error is detected if any. Both error detection and correction mechanism is available under this methodology[22], [23].

d. SAVEPOINT ROLE BACK SCHEME

Watchdog timer is effective mechanism used in order to detect crash, infinite loop and failure. Watchdog timer is reset many times during its operation. As the fault is detected timer is reset. If timer is not reset then system is turned off[24].

The hardware redundancy techniques are efficient enough to handle faults present within the system. But these techniques are expensive. The faults that can be detected are limited in nature. In order to tackle the problem of hardware and time redundant techniques hybrid approach is utilized rather than individual approach.

S.no.	Title	Year	Published By	Type of Fault handled
1	Fault Tolerant Approaches in Cloud Computing Infrastructures Alain[25]	2012	ICAS	Handled fault at different level application level, virtualization level and hardware level.
2	A Family of Fault-Tolerant Efficient Indirect Topologies[26]	2016	IEEE	Fault tolerance in network for high performance computing are done and it uses simple indirect topology to handle the faults
3	Fault Tolerance Management in Cloud Computing: A System-Level Perspective[27]	2016	IEEE	Relies on generic fault tolerance mechanisms that handle the fault at server end
4	Fault tolerance techniques and algorithms in cloud computing[28]	2014	IJCSCN	Handle the faults that enter in the system or software.
5	Optimising Fault Tolerance in Real-time Cloud Computing IaaS Environment[29]	2016	IEEE	Handle faults in real time computing on the cloud infrastructure.
6	Fault Tolerance in Cloud Using Reactive and Proactive Techniques 1Kalanirnika[30]	2015	IJCSEC	Manage faults in memory and perform recovery using checkpoints

7	Fixed-Priority Allocation and Scheduling for Energy-Efficient Fault Tolerance in Hard Real-Time Multiprocessor Systems[31]	2008	IEEE	Manage faults in hard real time systems using optimistic fault tolerance algorithms
8	On Fault Tolerance in Data Centre Network Virtualization Architectures [32]	2013	IEEE	Handle faults in Virtual Data Centres using the address handling techniques at server end.
9	Fault tolerance and QoS scheduling using CAN in mobile social cloud computing[33]	2013	Springer	Handle faults in mobile devices in computing environment and use CAN structure for fault management
10	A Performance Study of Deployment Factors in Wireless Mesh Networks[34]	2007	IEEE	Manage the faults that occurred in mesh topology during the connectivity and also gives the mechanism

CONCLUSION

Fault tolerance along with energy efficiency is the need of the hour. The proposed work provides detailed description of various approaches used to establish the same. Energy and reliability has trade off. This means that when energy consumption is high reliability is low and vice versa. Described approaches including time redundant and hardware redundant approaches are not efficient enough to tackle energy efficiency along with fault tolerance. In future hybrid approach could be area of analysis including shadow replication approach. Hybridization with check pointing can provide efficient energy efficiency fault tolerance within advance computing model.

REFERENCES

[1] R. Prodan and S. Ostermann, “A Survey and Taxonomy of Infrastructure as a Service and Web Hosting Cloud Providers,” no. November, 2009.

[2] F. International and J. Conference, “2009 Fifth International Joint Conference on INC , IMS and IDC,” pp. 44–51, 2009.

[3] Q. Zhang, L. Cheng, and R. Boutaba, “Cloud computing : state-of-the-art and research challenges,” pp. 7–18, 2010.

[4] M. Nazari, A. Khadem-zadeh, and M. Haghparast, “A Survey of Fault Tolerance Architecture in Cloud Computing Journal of Network and Computer Applications A survey of fault tolerance architecture in cloud computing,” *J. Netw. Comput. Appl.*, vol. 61, no. October 2017, pp. 81–92, 2015.

[5] G. Aceto, A. Botta, W. De Donato, and A. Pescapè, "Cloud monitoring : A survey," *Comput. NETWORKS*, 2013.

[6] W. Lang and J. Naughton, "Computer Sciences," no. March, 2010.

[7] A. M. Sampaio and J. G. Barbosa, "Towards high-available and energy-efficient virtual computing environments in the cloud," *Futur. Gener. Comput. Syst.*, vol. 40, pp. 30–43, 2014.

[8] C. N. H. G. Karagiannis, "Cloud computing services : taxonomy and comparison," pp. 81–94, 2011.

[9] T. Mastelic *et al.*, "Cloud computing : survey on energy efficiency To cite this version : HAL Id : hal-01153714," 2015.

[10] S. Fu and C. Xu, "Exploring Event Correlation for Failure Prediction in Coalitions of Clusters."

[11] A. Kumar, "An Efficient Ch heckpointing Approach h for Fault Tolerance in Tim me Critical Systems wi ith Energy Minimization," pp. 704–707, 2015.

[12] Z. Zhu, G. Zhang, M. Li, and X. Liu, "Evolutionary Multi-Objective Workflow Scheduling in Cloud," *IEEE Trans. Parallel Distrib. Syst.*, vol. 27, no. 5, pp. 1344–1357, May 2016.

[13] B. Mills, T. Znati, and R. Melhem, "Shadow Computing: An energy-aware fault tolerant computing model," *2014 Int. Conf. Comput. Netw. Commun.*, pp. 73–77, 2014.

[14] D. Burlyaev, P. Fradet, and A. Girault, "Time-redundancy transformations for adaptive fault-tolerant circuits," in *2015 NASA/ESA Conference on Adaptive Hardware and Systems (AHS)*, 2015, pp. 1–8.

[15] "10.1109@FTCS.1989.105616.pdf." .

[16] J. Devale, "Checkpoint / Recovery Overview : Checkpointing - Recovery," *Memory*, 1999.

[17] J. Hursey, J. M. Squyres, T. I. Mattox, and A. Lumsdaine, "The Design and Implementation of Checkpoint / Restart Process Fault Tolerance for Open MPI ∗."

[18] P. A. Lee and T. Anderson, "Fault Tolerance," Springer Vienna, 1990, pp. 51–77.

[19] E. Dubrova, "System-on-Chip - Hardware redundancy."

[20] K. Echtle, "Fault Tolerance based on Time-Staggered Redundancy," Springer Berlin Heidelberg, 1987, pp. 348–361.

[21] K. E. Grosspietsch, "Schemes of dynamic redundancy for fault tolerance in random access memories," *IEEE Trans. Reliab.*, vol. 37, no. 3, pp. 331–339, 1988.

[22] W. Krings and F. S. Sequence, "Hardware redundancy –," pp. 1–25, 2007.

[23] M. S. Adeghi, H. S. Oltani, and M. K. Hayyambashi, "The study of hardware redundancy techniques to provide a fault tolerant system," vol. 36, 2015.

[24] J. R. Barnes, "Watchdog Timers," in *Robust Electronic Design Reference Book*, Boston, MA: Springer US, 2004, pp. 860–867.

[25] A. Tchana, L. Broto, and D. Hagimont, "Fault Tolerant Approaches in Cloud Computing Infrastructures," no. c, pp. 42–48, 2012.

[26] D. F. Bermudez Garzon, C. G. Requena, M. E. Gomez, P. Lo, and J. Duato, "A Family of Fault-Tolerant Efficient Indirect Topologies," *IEEE Trans. Parallel Distrib. Syst.*, vol. 27, no. 4, pp. 927–940, Apr. 2016.

[27] A. S. Perspective, R. Jhawar, G. S. Member, and V. Piuri, "Fault Tolerance Management in Cloud Computing :," pp. 1–10, 2012.

[28] Y. L. Devi, "FAULT TOLEREANE TECHNIQUES AND ALGORITHMS IN CLOUD Cloud system," vol. 4, no. 1, pp. 1–8.

[29] B. Mohammed, M. Kiran, I. Awan, and K. M. Maiyama, "Optimising Fault Tolerance in Real-time Cloud Computing IaaS Environment," pp. 363–370, 2016.

[30] G. R. Kalanirnika and V. M. Sivagami, "Fault Tolerance in Cloud Using Reactive and Proactive Techniques," vol. 3, no. 3, pp. 1159–1164, 2015.

[31] R. M. Systems, T. Wei, P. Mishra, K. Wu, and H. Liang, "Fixed-Priority Allocation and Scheduling for Energy-Efficient Fault Tolerance in Hard," vol. 19, no. 11, pp. 1511–1526, 2008.

[32] S. C. Joshi and K. M. Sivalingam, "On Fault Tolerance in Data Center Network Virtualization Architectures," pp. 1–6, 2013.

[33] S. Choi, K. Chung, and H. Yu, "Fault tolerance and QoS scheduling using CAN in mobile social cloud computing," 2013.

[34] J. Robinson and E. W. Knightly, "A Performance Study of Deployment Factors in Wireless Mesh Networks," *IEEE INFOCOM 2007 - 26th IEEE Int. Conf. Comput. Commun.*, pp. 2054–2062, 2007.