IJESRT
INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES & RESEARCH TECHNOLOGY

FAULT HANDLING IN CLOUD: A BRIEF REVIEW

Moin Hasan [*1], **Major Singh Goraya** [2]
[*] Department of CSE, Sant Longowal Institute of Engineering and Technology, India

DOI: 10.5281/zenodo.839121

ABSTRACT

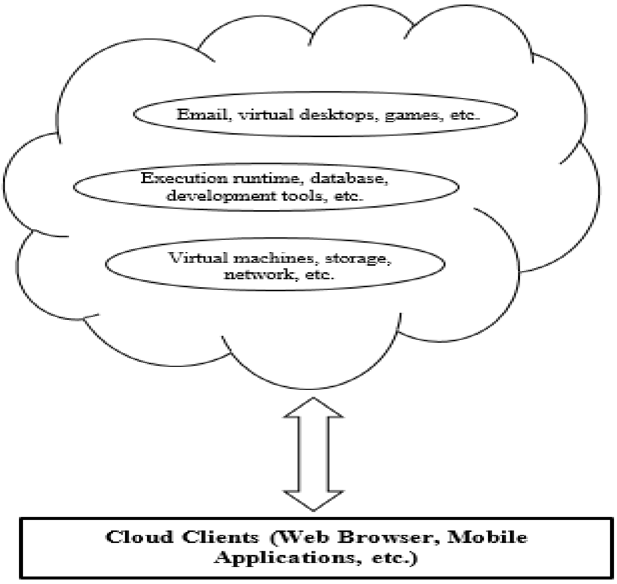
Cloud computing is an internet based paradigm which provide different computing services to millions of end users. Due to its complex distributed architecture and characteristics like dynamism and openness, it is always susceptible to faults and failures. Occurrence of faults in cloud delays the service delivery and consequently degrades the system performance. Therefore, an efficient and robust fault handling technique is always required to maintain the system reliability. Various fault handling techniques have been evolved through the years. This paper explains the various causes of faults and uncertainties and presents a brief survey on different fault handling approaches in cloud computing along with the techniques based on those approaches.

KEYWORDS: cloud computing; fault; fault handling; uncertainty; reliability

I. INTRODUCTION

NIST defines cloud computing as a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources [1]. These resources include *storage*, *processing power*, *networks*, *applications*, etc. The delivery of these resources is broadly classified as *infrastructure*, *platform*, and *software* as the services. Fig. 1 shows the brief architecture of cloud computing. The important characteristics of cloud computing which attract scientific as well as commercial organizations to acquire it are *decentric control*, *on-demand access*, *rapid elasticity*, *resource autonomy*, *'Always-on' availability*, etc. [2]. Besides this, certain characteristics like openness, dynamism, etc. makes cloud computing a failure prone environment. Moreover, its complex distributed architecture further makes the task of handling the faults and failures challenging [3].

Figure 1:


Email, virtual desktops, games, etc.
Execution runtime, database, development tools, etc.
Virtual machines, storage, network, etc.
Cloud Clients (Web Browser, Mobile Applications, etc.)

Cloud Computing Architecture

Generally, occurrence of faults creates the differences between the normal and actual behavior of the system. It delays the service delivery and affects various parameters like *performance*, *bandwidth*, *processing time*, *reliability*, etc. [4]. Sometimes, the effects of faults are so adverse that the economical state of the service provider is traumatized. For example, in 2013, Amazon was down for just 45 minutes due to an unexpected fault, which caused an economic loss of $5 million [5]. Even after fetching so much attention, cloud computing has not yet reached the level of maturity expected by its customers [6]. Therefore, fault handling is considered an open challenge in cloud computing. Following contributions are made in this paper:

- To explain various causes of faults in cloud computing.
- To acquaint with different fault handling approaches in cloud computing.
- To explain various techniques based on existing fault handling approaches.

Rest of the paper is organized as follows: Next section gives the definition of fault and explains its types and causes in cloud computing environment. Different fault handling approaches along with the techniques respectively based on them are introduced in section III. A brief discussion is made in section IV and the paper is concluded in section V.

II. TYPES AND CAUSES OF FAULTS IN CLOUD

Different types of faults have been observed in literatures through extensive researches which are associated with cloud computing. Some of them are briefly described with their causes as follows [4], [5]:

- Parametric faults: The faults occurred due to the unknown variation in the parameters are called parametric faults.
- System faults: The faults occurred due to incomplete knowledge of the processes that control the service provisioning in the system are called system faults.
- Configuration faults: These faults occur when the ordering of the system components is disturbed.
- Software faults: These faults are generally the resultants of software updates in the system. For example, if an upgrade has taken place overnight and a key feature that you relied on has been removed.
- Hardware faults: These faults occur at the infrastructure level in the service delivery model of the cloud system. They are caused due to the failure of any hardware component.
- Resource contention faults: These faults are the resultants of the conflict when a resource is being shared for the access.
- Stochastic faults: Due to insufficient statistical information to assess the system state, the calculation of probability of fault occurrence becomes difficult. The faults occurred in such cases are called stochastic faults.
- Participant faults: These faults occur due to the conflict between cloud participants like consumer, provider, administrator, etc.
- Constraint faults: When a fault condition arises, and ignored by the responsible agent, the faults occurred at such instance are called constraint faults.
- Retrospective faults: The faults occurred due to the lack of information about the past behavior of the system are called retrospective faults.

Apart from the above discussed faults, cloud is also susceptible to *virtualization faults*, *migration faults*, *elasticity faults*, etc.

III. FAULT HANDLING APPROACHES IN CLOUD

Fault handling approaches in cloud are broadly classified as *proactive* and *reactive*. In proactive approaches, provisions are taken so that faults would not occur. These approaches generally use the concepts of Artificial Intelligence like, *neural networks*, *fuzzy logic*, etc. for the pre-learning of the system. The basic proactive fault handling approaches are *fault forecasting* and *fault prevention* [7]. Some important researches based on proactive fault handling approaches in cloud are given as follows:

- **Wang** *et al.* in **2015** proposed an online incremental clustering technique to diagnose faults for web applications in cloud computing [5]. Online incremental clustering is used to capture workload fluctuations. The faults at an instance are detected by modeling the correlation between the workloads and the application performance metrics. In order to model the correlation, canonical correlation analysis is used. If there is an abrupt change in the correlation coefficients, it is considered as a fault.
- **Sood** and **Sandhu** in **2015** proposed an adaptive proactive approach towards the resource provisioning in mobile cloud environments [8]. They introduce two-dimensional resource provisioning matrices in

which the usages of the resources are stored. These matrices are used through neural networks in order to predict the future.

- **Sampaio** and **Barbosa** in **2014** proposed a scheduling strategy to execute sets of independent tasks in cloud computing [9]. The proposed strategy consists of two components, viz. *cloud manager* and *cloud scheduler*. The authors consider each set of tasks as job and an arrived job is first given to the *cloud manager*. Inside the *cloud manager*, there is a *condition detector*, which examines each task in the job. On the basis of this examination, the *physical machine* for the task is assigned first and then the *virtual machine*. The task is then scheduled on the assigned *VM*. In case of a failure prediction, a *VM* can be migrated as per the *stop-and-copy* approach.

In reactive approaches, measures are taken in order to handle the faults after their occurrence. The basic reactive fault handling approaches are *fault removal* and *fault tolerance* [7]. Fault removal approach removes the faults by executing the system maintenance programs. On the other hand, fault tolerance approach uses the concept of redundancy for its applicability. It means that multiple resources are assigned to execute a single task. *Primary-backup* and *task replication* are the two renowned techniques based on redundancy [10]. In primary-backup technique, a backup resource is provided. Task is scheduled on the primary resource. If primary resource fails, task is migrated to the backup resource [11]. In task replication technique, same task is scheduled on multiple resources which execute the task in parallel [12]. Following are the renowned researches based on reactive approaches:

- **Hasan** and **Goraya** in **2017** proposed a fault tolerant computing service framework with better resource utilization [12]. They customized the previously proposed framework (Cooperative Computing System [13]) for Cloud environment. The proposed framework is well capable of executing the primary tasks within the specified deadlines, while the resource utilization is improved by executing the secondary tasks.
- **Wang** *et al.* in **2015** presented a fault tolerant scheduling mechanism for real-time tasks in virtualized clouds [14]. They utilized the primary backup approach for the fault tolerance. In the proposed mechanism, the users' tasks are queued in an input buffer and then transferred to the scheduler, which has three basic components viz. *resource controller*, *backup copy controller*, and *real-time controller*. As per the agreement of these components, each task is scheduled on two different virtual machines lying in different hosts (primary and backup). At the arrival of a new task, the two hosts are vertically scaled up in order to provide a new virtual instance to the arrived task. Similarly, in case of task departure, the hosts are vertically scaled down.
- **Chen** *et al.* in **2015** proposed a fault-tolerant framework for data storage and processing in dynamic clouds [15]. For the purpose, they integrated the concept of *k-out-of-n* mechanism from distributed computing into cloud computing. In order to store and process data, two functions are developed, namely *AllocateData()* and *ProcessData()* respectively. The methodology first separates the storage requests and processing requests and passes them to their respective functions. The probability of operation failure is then estimated, on the basis of which the expected transmission time is computed. After that, the *k-out-of-n* mechanism is applied and the resources are finally allocated.
- **Jhawar** *et al.* in **2013** proposed a fault tolerant management framework in IaaS model of cloud computing [16]. The fault handling is supposed to be done by the third party contracted by the cloud provider and the framework is named as *fault tolerance as a service*. The fault tolerance is applied at the virtualization layer directly rather than at the application being deployed, by replicating the whole virtual instance. Faults are detected by a run-time monitoring system which uses heartbeat protocol. The primary component periodically sends a liveness request to all the replicated backups. A timer is maintained for each request. If the replicated backup fails to respond *N* liveness requests within a predefined time, it is considered to be failed.
- **Sun** *et al.* in **2012** modeled a fault-tolerant serviceability in cloud computing environments using the check-pointing technique [17]. The authors first analyzed the mathematical relationship between the different failure rates and check-pointing strategy and then developed a model to provide fault-tolerant services in cloud named as DAFT (Dynamic Adaptive Fault Tolerance).
- **Malik** and **Huet** in **2011** presented a fault tolerant real-time tasks' execution model in cloud computing [18]. The real-time incoming tasks are maintained in an input buffer. Tasks in FCFS manner are then promoted for execution. Each task is replicated on *M* virtual machines, which are embedded with different algorithms for real-time task execution. The result produced by each algorithm is moved further for the acceptance test, where the correctness of the result is verified. The results are then moved to the

time checker so as to check whether the result is obtained before deadline or not. If none of the results is obtained before deadline, the task is sent back to input buffer. Based on the obtained results, the reliabilities of the corresponding virtual machines are adjusted.

IV. DISCUSSION

The advantage of proactive fault handling approaches is that they mask the faults that could possibly occur. It helps in providing seamless service delivery to the end users and consequently improves the system performance. However, proactive approaches can successfully mask the faults which arise due to the internal conditions only. The faults which arise due to the external conditions are generally unavoidable [19]. Therefore, it confines the applicability of proactive approaches. Moreover, proactive approaches are based on Artificial Intelligence concepts as already discussed in section III, therefore, they may lead to severe penalties for any imprecise detection.

On contrary, cloud possesses the characteristic of highly available on-demand resources [3], [20]. So, the reactive approaches can take this characteristic as an advantage as they use the concept of resource redundancy. The basic technique which uses redundancy is *task replication* as discussed earlier. The *primary-backup* technique is a variation of task replication. Task replication is further classified as *semi-active* and *semi-passive* [21]. In semi-active replication, both primary and backup resources perform the given operation simultaneously. In case of primary resource failure, any of the backup resources is assigned as primary. In semi-passive replication, only the execution updates are given to the backup resources and primary resource performs the operation. If primary resource fails, any of the backup resources is assigned as primary and execution is resumed on this resource from the last updated state. The main disadvantage of replication is that it has high communication overhead [13].

V. CONCLUSION

Cloud computing has been the most acquired paradigm of computing service provisioning for the last decade. Even though, it possesses numerous characteristics that attract even low level organizations as well individual users, but due to its complex and failure prone architecture, it is still considered as an infant computing paradigm. In this paper various types of faults in cloud computing environment have been discussed. Different fault handling approaches along with the techniques based on them have also been discussed.

VI. ACKNOWLEDGEMENTS

The authors would like to thank Sant Longowal Institute of Engineering and Technology (Sangrur, India) for the research fellowship.

VII. REFERENCES

[1] P. Mell and T. Grance, “The NIST definition of cloud computing,” *National Institute of Standards and Technology*, 2011. [Online]. Available: http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf. [Accessed: 01-Sep-2016].

[2] M. Hasan and M. S. Goraya, “Priority based cooperative computing in cloud using task backfilling,” *Lect. Notes Softw. Eng.*, vol. 4, no. 3, pp. 229–233, 2016.

[3] D. Puthal, B. P. S. Sahoo, S. Mishra, and S. Swain, “Cloud Computing Features, Issues, and Challenges: A Big Picture,” *2015 Int. Conf. Comput. Intell. Networks*, pp. 116–123, 2015.

[4] A. Tchernykh, U. Schwiegelsohn, V. Alexandrov, and E. Talbi, “Towards understanding uncertainty in cloud computing resource provisioning,” in *Proceedings Internatinal Conference on Computational Science*, 2015, vol. 51, pp. 1772–1781.

[5] T. Wang, W. Zhang, C. Ye, J. Wei, H. Zhong, and T. Huang, “FD4C: automatic fault diagnosis framework for web applications in cloud computing,” *IEEE Trans. Syst. Man, Cybern. Syst.*, vol. 46, no. 1, pp. 61–75, 2016.

[6] M. Tebaa and S. EL Hajji, “From single to multi-clouds computing privacy and fault tolerance,” in *Proceedings International Conference on Future Information Engineering*, 2014, vol. 10, pp. 112–118.

[7] W. Qiu, Z. Zheng, X. Wang, X. Yang, and M. R. Lyu, “Reliability-based design optimization for cloud migration,” *IEEE Trans. Serv. Comput.*, vol. 7, no. 2, pp. 223–236, 2014.

[8] S. K. Sood and R. Sandhu, "Matrix based proactive resource provisioning in mobile cloud environment," *Simul. Model. Pract. Theory*, vol. 50, pp. 83–95, 2014.

[9] 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.

[10] M. Hasan and M. S. Goraya, "A framework for priority based task execution in the distributed computing environment," in *Proceedings IEEE International Conference on Signal Processing, Computation and Control*, 2015, pp. 155–158.

[11] Q. Zheng, B. Veeravalli, and S. Member, "On the Design of Fault-Tolerant Scheduling Strategies Using Primary-Backup Approach for Computational Grids with Low Replication Costs," vol. 58, no. 3, pp. 380–393, 2009.

[12] M. Hasan and M. S. Goraya, "Resource efficient fault-tolerant computing service framework in cloud," *Int. J. Comput. Sci. Eng.*, vol. 9, no. 3, pp. 51–60, 2017.

[13] M. S. Goraya and L. Kaur, "Fault tolerance task execution through cooperative computing in grid," *Parallel Process. Lett.*, vol. 23, no. 1, pp. 1–20, 2013.

[14] J. Wang, W. Bao, X. Zhu, T. Yang, and Y. Xiang, "FESTAL: fault-tolerant elastic scheduling algorithm for real-time tasks in virtualized cloud," *IEEE Trans. Comput.*, vol. 64, no. 9, pp. 2545–2558, 2015.

[15] C.-A. Chen, M. Won, R. Stoleru, and G. G. Xie, "Energy-efficient fault-tolerant data storage and processing in mobile cloud," *IEEE Trans. Cloud Comput.*, vol. 3, no. 1, pp. 28–41, 2015.

[16] R. Jhawar, V. Piuri, and M. Santambrogio, "Fault tolerance management in cloud computing: a system-level perspective," *IEEE Syst. J.*, vol. 7, no. 2, pp. 288–297, 2013.

[17] D. Sun, G. Chang, C. Miao, and X. Wang, "Modelling and evaluating a high serviceability fault tolerance strategy in cloud computing environments," *Int. J. Secur. Networks*, vol. 7, no. 4, pp. 196–210, 2012.

[18] S. Malik and F. Huet, "Adaptive fault tolerance in real time cloud computing," in *Proceedings - IEEE World Congress on Services*, 2011, pp. 280–287.

[19] A. Abid, M. T. Khemakhem, S. Marzouk, M. Ben Jemaa, T. Monteil, and K. Drira, "Toward antifragile cloud computing infrastructures," *Procedia Comput. Sci.*, vol. 32, pp. 850–855, 2014.

[20] B. K. Rani, B. P. Rani, and A. V. Babu, "Cloud Computing and Inter-Clouds – Types, Topologies and Research Issues," *Procedia Comput. Sci.*, vol. 50, pp. 24–29, 2015.

[21] W. Zhao, P. M. Melliar-Smith, and L. E. Moser, "Fault tolerance middleware for cloud computing," in *Proceedings IEEE 3rd International Conference on Cloud Computing*, 2010, pp. 67–74.

CITE AN ARTICLE

Hasan , Moin , and Goraya Singh, Major. "FAULT HANDLING IN CLOUD: A BRIEF REVIEW." *INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES & RESEARCH TECHNOLOGY* 6.8 (2017): 74-78. Web. 5 Aug. 2017