RESEARCH REVIEW JOURNALS
Volume-04
Issue-01
January -2019

ISSN: 2455-3085 (Online)
RESEARCH REVIEW International Journal of Multidisciplinary
www.rrjournals.com [UGC Listed Journal]

A Study of Reliability of Cloud Proactive Fault Tolerance Strategy

[1]Tarun Mahajan and [2]Dr. Jitendra Sheetlani

[1]*Research Scholar, Sri Satya Sai University Of Technology & Medical Sciences, Sehore*
[2]*Associate Professor,Sri Satya Sai University Of Technology & Medical Sciences, Sehore*

ARTICLE DETAILS

Article History
Published Online: 20 January 2019

Keywords
Reliability, Cloud Proactive, Fault Tolerance, Cloud computing, Information Technology, power management software.

ABSTRACT

The primary component of information technology is cloud computing. It offers mobility, scalability, elasticity and low computational cost to skyscrapers. Such cloud features have helped companies switch their legacy systems to the cloud world. Although it has many advantages, it becomes extremely difficult to achieve high reliability during migration. Reliability is thus considered to be the essential prerequisite to move applications to the cloud world. - critical component's failure tolerance capability is evaluated using different strategies, such as Recovery Block, NVP and Parallel. The efficiency of the program is increased by optimizing a limited number of error-prone elements. Cloud computing is a method for providing customer service. It provides on request services and is thus known as a generic "pay per use." Cloud features include self-healing, multi-purpose, SLA powered, linearly scalable, virtualized and flexible. It made the customers think in all directions. It helps the provider to automatically reconfigure the resources on the basis of service level agreements. The recent "Quality in the Cloud" survey reveals that research and growth, management and operation are the leading operating areas for cloud adoption. Through a detailed assessment involving a systematic review of current applications, a multi-dimensional analysis of the application scope, an examination of the cloud readiness of each application, a much-scaled approach can be developed to suit the cloud model for entrepreneurship. Once held in a definition, the specification is operationally unreliable and often amended. Energy management tools, automatic replication tools, and triggers live migration are methods for increasing cloud reliability.

1. Introduction

Across different areas of cloud computing, customers and businesses are gradually realizing the importance. When the industry matures, it tackles health, stability and pricing issues more effectively. This would lead to further industry-wide adoption. In any way possible, the machine travels easily to the cloud. This will be a long transition, considering that only once connectivity is everywhere is cloud computing truly ubiquitous. These cloud features have led companies to migrate their conventional cloud applications. Nevertheless, migration should not degrade the application 's performance. Therefore, the efficiency of applications in the migration phase needs to be strengthened. The naming of critical components to be transferred into the cloud environment would increase the reliability.

Huang and Abraham published the defective tolerant matrix operation in 1984, which suggested an Algorithm- Based Fault Tolerance (ABFT) method. Mei et al .(2008) have explored the problems of dynamic registration, massive data storage and access, adaptability and cloud quality discovery. Cachin et al .(2009) published a survey of trust and privacy in cloud computing. The survey focused on data storage in cloud grids and how anonymity, fairness, reliability and service guarantee can be accomplished. Chakrabarti et al. spoke about grid computing security problems. Since clouds can be created by multiple grids from various organizations, security problems such as confidentiality , integrity and availability of services must be addressed. Cloud computing was also introduced for the safety of customers. In preparing data for cloud storage, Wang et al .(2006) proposed to use erasure correction coding.

Cloud computing involves operating programs that can be distributed geographically on different virtual servers. Such virtual servers are designed to solve various service level agreements and reliability problems. There are several virtual server instances. Every cloud-based program is replicated. You have access to various parts of the hardware. If one component fails, it is replaced directly with another. It thus preserves the available information and easily recovers from the loss.

The offered services can be divided into three main categories. These are the services of tech, the operation of network and infrastructure. The technology that software as a service provides customers with applications on their order. It has one cloud service instance and provides multiple end users with a service. The customer needs no upstream investments as servers; licenses are also reduced while the side expense of the company is reduced. Microsoft, Zoho, Twitter, Salesforce, etc are the best examples for this kind of operation. Platform as a Service is the second category of service in cloud computing. This kind encapsulates and provides a service the development environment or software layer. In the form of the lower level layer the higher layer may successfully be constructed. The PaaS providers include Ubuntu, PHP , MySQL, and apache, a pre-defined mix of OS and applications. The Google App Engine is the perfect example of this program. The customer is free to create his own apps that operate on the network of the provider. The Service Infrastructure offers basic storage and computing as structured network services. The workload is managed and shared by servers , storage facilities, networking equipment, datacenter space etc. Usually, the customer will implement their own network applications. Amazon, 3 Tera, GoGrid, etc. are common examples.

2. Literature Review

Papaioannou et al. (2010) Propose a self-managed key value store that distributes data storage services efficiently and cost-effectively to many applications. Our methodology provides multiple distinct availability guarantees for each application despite failures and maintaines them dynamically. Our virtual economy is focused upon the option of copying, duplicating, or removing each partition of data (i. e. a key range in a regularly hazardous room) focused on the benefit maximization of the value of the partition, and the cost of storing and maintaining it. As a game-theoretical model has shown, the system does not have migrations or replications in

the balance which is soon arrived upon stability of query load and storage. We have also shown by means of comprehensive experiments in simulation that our approach determines dynamically the optimal allocation of resources that matches the overall database processing cycle and achieves cost-effective availability goals for different database and storage levels. We have finally introduced a fully operational prototype of our strategy, demonstrating its applicability in real environments clearly.

HORNING et al. (1974) describes a method of structuring programs which aids the design and validation of facilities for the detection of and recovery from software errors. Associated with the method is a mechanism for the automatic preservation of restart information at a level of overhead which is believed to be tolerable.

Gokhale et al. (2002) Build a detailed hierarchical model to predict the performance and reliability, based on its design, of component software systems. This model reflects the difference in the number of visits to each module and thus provides predictions that are similar to those provided by a composite model. The methodology built in this paper helps to recognize bottlenecks in performance and reliability. We also create expressions for evaluating the performance sensitivity and reliability predictions for changes in individual modules parameters. However, we demonstrate how the hierarchy model is used to test the impact of changes in workload on application efficiency and reliability. We demonstrate the prediction of results, reliability and sensitivity analysis with examples.

Hecht et al. (1976) proposes and evaluates a fault tolerance framework for supporting the development of dependable applications. This framework is build upon basic operating system services and middleware communications and brings flexible and transparent support for application threads. A case study involving radar filtering is described and the framework advantages and drawbacks are discussed.

Khajeh et al. (2011) Study reveals possible advantages and threats from the company-wide spectrum of stakeholder viewpoints from in-home data center to Amazon EC2, transcending the traditional but limited financial and technological analysis provided by service providers, correlated with the conversion of an IT infrastructure from the oil and gas industry to Amazon EC2. Our results indicate that the network infrastructure will cost EC2 37 percent less over five years in the event of a study and 21 percent of support calls could theoretically have been avoided using cloud computing. Such results seem to be relevant enough to call for the program to move to the cloud, but we found that substantial risks are correlated with our stakeholder effect study. Whilst the advantages of cloud usage are enticing, it is important to take company decision-makers into account the overall organizational impacts of cloud computing changes in order to prevent local excitement from being introduced at the expense of organizational efficiency.

Zheng et al. (2012) Propose an FTCloud feature ranking system for fault tolerant cloud applications. Two ranking algorithms are used in FTCloud. The first algorithm uses structures for invoking components and frequencies for important component rating. The second classification algorithm systematically incorporates the details on the network layout and the expertise of application designers to classify the essential components of the cloud application. After the component classification process an algorithm is proposed that an optimum fault tolerance strategy is automatically developed for the major cloud components. Experimental findings show that the reliability of cloud systems can be significantly enhanced by tolerating defects in a limited number of the most critical components.

3. Fault tolerance in cloud computing

As a succeeding new paradigm in the IT industry, cloud computing grew. Cloud computing can be described as simply organizing and providing resources , information, software and applications as cloud-enabled services. Cloud dynamics are more or less vulnerable to failure. This is the accepted platform for information and services integration. Failure can be measured and successfully handled to achieve stability in cloud computing. Fault tolerance requires policies and strategies allowing a system to continue operating or process execution correctly or to fix faults if any of its components are failing. This paper uses a constructive technique of fault tolerance. We recommend the use of the Naïve Bayes classification to identify the nodes as fault-sensitive and those that are not and use proactive techniques of fault tolerence. Reliability is achieved with the use of the Naïve Bayes system and the defect tolerance techniques. Cloud computing is a concept to allow the universal access across the Internet to shared groups of configurable resources. Services are offered across the Internet in cloud computing by the use of various abstraction levels and the implementation of different models. The cloud computing term is known as the distributed service of different software to individuals, ranging from a personal (small) domain , for example, to wide public domains as any organization providing the IT infrastructure to external data centers. Google's Gmail is an example of cloud-based services; a broad rank of cloud-based services and solutions is provided by several major companies including Google, Intel, IBM, Amazon, Oracle etc. Although cloud computing and its services are increasingly being implemented in the industry today, many serious topics still require our attention and our research, including fault tolerance in the cloud , cloud health, workflow programming, etc. Management of default tolerance is seen as one of the most critical issues to be tackled entirely.

The failure of the cloud system is caused by an error caused by the system fault. Fault tolerance is used to avoid or cope with machine faults following development, or to deal with those arising during execution of the software fault. Fault tolerance In cases of failure, the operating framework provides the mechanism to handle and manage the failures in order to avoid system failure and guarantee the continuity of services. The key benefits that we reap are regenerative failures, improved performance monitoring and cost savings by introducing fault tolerance in the cloud. The purpose of this article is to provide a better understanding of cloud computing and to address algorithms and techniques used in the field of defect tolerance. This paper also contains our own proposed cloud computing defect tolerance model.

4. Fault Tolerance Strategies

Reactive fault tolerance strategies reduce the effect of failures on application execution, these techniques help in troubleshooting our cloud computing system on account of failure occurrences. Based on these policies there are various techniques listed as under.

1. **Check pointing/ Restart –** This technique is very efficient for long running applications. In this task level fault tolerance technique, whenever some task failure occurs, it is provisioned to restart from a state which was most recently checked.
2. **Task Resubmission –** The task resubmission strategy is used broadly in the fault tolerance domain. In this technique, whenever we detect a failed task, we resubmit that task to the same resource or to some dissimilar resource.
3. **Job Migration –** In this technique, whenever a task failure occurs, that failed task is migrated to an alternative machine and this process is termed as job migration. This fault tolerance strategy can be deployed using the HAProxy software.

4. **User defined exception handling –** In user defined exception handling reactive fault tolerance technique, users themselves lay down the treatment or response to a failed task. [1]
5. **Replication –** We can make use of the replication strategy by deploying tools such as the HAProxy or Hadoop or the AmazonEc2 etc. What actually happens in this strategy is that many replicas of a particular task run simultaneously on different platforms for the successful completion of task until all the replicated tasks are crashed.
6. **Retry –** Retry is one of the simplest technique used in reactive fault tolerance domain. It, again and again, retries the tasks that fail arranged on the same cloud platform to eliminate the cause of failure when a fault occurs in the cloud environment.
7. **SGuard –** The SGuard technique is based on rollback recovery. SGuard is not as much of troublesome to usual stream dispensation. More resources are available in SGuard strategy. This method can be deployed in Amazon EC2 etc. [1]
8. **Rescue workflow –** The rescue workflow reactive fault tolerance strategy allows the process execution to continue even if task failure occurs in between and this policy continues up to the point where for the process it is no longer possible to proceed further without mending or recovering the task that failed in between. [1]

5. Proactive Fault Tolerance

The Proactive Fault Tolerance strategies involve predicting the faults, errors etc. and proactively replacing the suspected components that are most prone to failure with the alternative components that are in working condition. Some of the methodologies based on the proactive fault tolerance concept are listed as under –

1. **Software Rejuvenation –** The Software Rejuvenation is a proactive fault tolerance technique in which a system is designed in a way for a regular periodic reboot and the system is then restarted with a new clean state. The main problem with this technique is if it is performed too often the cost hoards unnecessarily. Also, if the system is designed to rejuvenate at very long intervals there's a prominent risk that it might fail, the ideal rejuvenation interval would always end just before the cloud system fails.
2. **Proactive Fault Tolerance using Self-Healing –** The Self-healing technique has the feature of handling the failure of application instances automatically when numerous instances of the application run on several VMs.
3. **Proactive Fault Tolerance using Preemptive Migration –** In proactive fault tolerance using preemptive migration, we preemptively drift away the currently executing application from the nodes which are about to fail. The process execution is constantly monitored throughout the cycle. While the application is being migrated, first its state is saved and then migration to a different node takes place.

6. Conclusion

In a way, if these complex cloud computing environments are more or less vulnerable to failure, successful failure tolerance approaches will guarantee reliability across these cloud systems, because their failure could be catastrophic. The cloud computing industry can only grow according to current trends. We worked with this proposed model in the proactive fault management area, predicting first the nodes most likely to encounter failure with Naïve Bayes classification and then using defect tolerance techniques to ensure improved network reliability. We may improve system reliability by using our proposal model, reliability values to estimate the risk of failure of any variable, such as our node. The machine efficiency is calculated by the factor MTBF. For Naïve Bayes, the number of node failures can be reduced by almost 87 percent with precision. Many classification algorithms can be used for the same and then compared. In order to improve and reduce the overhead, scalability, which is a QOS feature, can also be checked. The plan aims to enhance the application 's stability during cloud migration. This requires extracting standardized information from applications, defining the coupling and calling frequency of components. If the number of frequencies and ratio of connections are small, the dependency between the components is greater. The created component graph would help to identify the relation between the components in the request. The failure rate of each part is determined by using the above information. When the part is compromised or fails, it will have a higher failure rate impact on the application. Therefore, the software efficiency of the application can be further improved by removing or refactoring these bottleneck components.

References

1. N. Bovin, T.G.Papaioannou, A self-organized fault tolerant and scalable replication scheme for cloud storage, *ACM symposium, USA,2010.*
2. HORNING, J. J., et al., "A Program Structure for Error Detection and Recovery," in E. Gelenbe and C. Kaiser (eds.), *Lecture Notes in Computer Science, Vol. 16, New York: Springer-Verlag, 1974.*
3. S.S. Gokhale, K.S.Trivedi, Reliability prediction and sensitivity analysis based on software architecture, *ISSRE, 2002.*
4. Hecht. H, "Fault Tolerant Software for Real- Time Applications," *ACM Computing Surveys, Vol. 8, No. 4, 1976.*
5. A.Khajeh, Hosseini, D.Greenwood, I. Sommerville, Cloud migration: A case study of migrating an enterprise IT system to Iaas, *IEEE 3rd International Conference, Cloud, 2011.*
6. Z. Zheng and M.R. Lyu, ''Component Ranking for Fault-Tolerant Cloud Applications,*" IEEE Trans. Serv. Comput.(TSC), vol. 5, no. 4, pp. 540-550, 2012.*
7. Gagne, M, 2007, 'Cooking with Linux—still searching for the ultimate Linux distro?', *Linux Journal*, vol. 09, no. 161.
8. Greenberg, A., Hamilton, J., Maltz, D., and Patel, P. (2009) 'The Cost of a Cloud: Research Problems in Data Center Networks,' *ACM SIGCOMM Computer Communication Review,* vol. 39, no. 1, pp. 68-79.
9. Bilal, K, Malik, S., Khan, S. U., &Zomaya, A. (2014) 'Trends and Challenges in Cloud Data Centers,' *IEEE Cloud Computing Magazine*, vol. 1, no. 1, pp. 10-20, 2014.
10. Bauer, E., & Adams, R. (2012). *Reliability and availability of cloud computing*: John Wiley & Sons.
11. Goiri, I, Julia, F, Guitart, J, & Torres, J, 2010, 'Checkpoint-based fault-tolerant infrastructure for virtualized service providers', *In IEEE Network Operations and Management Symposium (NOMS)*, pp. 455-462.
12. Hakkarinen, D, & Chen, Z, 2013, 'Multi-Level Diskless Checkpointing', *IEEE Transactions On Computers*, vol. 62, no.4, pp. 772-783.
13. Nicolae, B, &Cappello, F, 2011, 'BlobCR: efficient checkpoint-restart for HPC applications on IaaS clouds using virtual disk image snapshots', *ACM International Conference for High Performance Computing, Networking, Storage and Analysis*.

