A Study on Fault Tolerance Mechanisms in Cloud Computing

Mohammad H. Alshayeji*, Mohammad Al-Rousan, Eman Yossef, Hanem Ellethy
Computer Engineering Department, College of Computing Sciences and Engineering, Kuwait University P.O. Box No: 5969, Safat 13060, Kuwait.

* Corresponding author. Email: m.alshayeji@ku.edu.kw
Manuscript submitted August 6, 2017; accepted Sptember 22, 2017.
doi: 10.17706/ijcee.2018.10.1.62-71

Abstract: Cloud computing is widely popular due to its elasticity, economics, reliability and much more. Cloud computing offers a scalable service without any initial investment in servers, storages, or networks. Fault Tolerance (FT) is the ability of any system to continue performing its function regardless of any unexpected hardware or software failures. Fault Tolerance in Cloud Computing (FTCC) is an important area of research due to its complexity. However, there is a lack of studies in this field. Moreover, recent failures and availability issues in popular cloud providers demonstrates the need for more effective solutions. In this paper, we present a study on FTCC mechanisms and analyze its strength and weakness. Based on the study, a comparison on the main fault tolerance techniques is presented considering the cost, overhead, failure types, performance, and the tools used. Moreover, we study and compare the models that enhance the performance of checkpoint and replication based techniques.

Key words: Cloud Computing, checkpoint, fault tolerance, replication.

1. Introduction

Currently, cloud computing plays a significant role in providing computing services to businesses of varying sizes. Failures of cloud services can have substantial financial loss. In case of mission critical real time systems, it can be even disastrous [1]. To ensure higher reliability, cloud computing systems need to be exceptionally fault tolerant. Fault Tolerance (FT) is the ability of a system to deliver the desired services even when failures and errors happen in the system. Cloud computing has many advantages including service scalability and flexibility. However, the most common concerns from user's perspective are performance, reliability, control, security, and privacy. Cloud users expect services to be available at all time and have access to their data from any location.

Several outages have occurred in cloud provider's systems. For instance, recently, sales force a popular cloud application provider went down for a whole day [2]. Apple iCloud experienced an outage during which various services such as email, game center, and iTunes were impacted. Users were unable to access the affected services due to a failure in authentication. The services were restored only after couple of hours Dropbox, one of the largest cloud data providers, was also impacted. Dropbox users were unable to access their documents [3].

Failures and outages affected most popular cloud computing providers such as Samsung, Microsoft exchange, Verizon Wireless and others [4], [5]. Improved FT methods can significantly enhance the

reliability of cloud systems. As computing in cloud is performed at remote systems there are more possibilities of failures due to undetermined latency, and loss of control over computing node. Therefore, it is essential that remote systems are highly reliable. There is a lack of studies on Fault Tolerance in Cloud Computing (FTCC) techniques. Most of the current work only focuses on the definitions and introduce some techniques, or discusses toolkits that provide models to enhance the reliability of the system. Our study attempts to analyze the available fault tolerance techniques and models in cloud computing. We summarize the FTCC technique and models in two different tables to present their strengths and weakness.

The paper is organized as follow. Section 2 presents the related work. The study of fault tolerance techniques is presented in Section 3. Section 4 presents the comparative study. Finally, the conclusion is presented in Section 5.

2. Related Work

A survey on FTCC techniques is presented in [5]. The paper discussed and described the techniques, the tools used, the policies and the research obstacles. The paper classified the techniques into proactive and reactive based techniques. Checkpoint and replication are considered as reactive techniques while s-guard is determined as a proactive technique.

A comparison of FT techniques presented in [6], it has details of the available software to implement FT, the programming framework, environment, the detected faults, and the application types. In addition, they proposed an involuntary fault tolerance for cloud climate architecture based on High Availability Proxy (HAProxy) for its implementation. Their result showed that the proposed technique is capable of dealing with various faults in cloud computing server application. Nevertheless, the comparison is related only to the used tools to implement the FT techniques and its programming language.

A survey on FT and resilience in cloud computing environment is presented in [7]. The paper presented a view of the FT model based on the architecture of the cloud computing, the servers failure condition, and the network failure condition. In addition, they classified the existing FTCC techniques based on the ability to tolerate crash failures or byzantine failure.

Proactive and reactive FT techniques are presented in [8]. Reactive techniques start to work after an error occurred and has a negative impact on the system. However, proactive fault tolerance predicts the error before it occurs or affects the system. Fault Tolerance Manager (FTM) and Message Passing Interface (MPI) are the examples used to demonstrate the reactive FT technique. Self-healing and preemptive migration examples are used to illustrate the proactive FT techniques.

A Survey on FTCC is presented in [9]. The paper introduced simple efficient mechanism that handles various kinds of failures including crashes, omissions, and arbitrary failures. The survey discussed in detail the types of failures, and the various mechanisms used for finding, handling, and recovering from failures. Moreover, the paper presented an optimization mechanism that improves the FT and listed the challenges and open issues in designing efficient FT mechanisms. However, it was focused only on map reduce based systems.

3. Study of Fault Tolerance Techniques in Cloud Computing

3.1. Reliability and Failures

A report carried out by European Network and Information Security Agency (ENISA) [10] presented around twenty-three risks. They categorized the risks into three classes: policy and organizational, technical, and legal. A table for each risk contains the probability estimate, impact estimate, level of the risk. They presented risk assessment to assess the impact of the risk on business and measure the likelihood of the risk. As shown in Fig. 1 there are no risks that can be classified as low risk impact (0-2) so we removed

it from the figure, sixteen risks are considered as medium risk (3-5) and nine risks are considered as high risk (6-8).

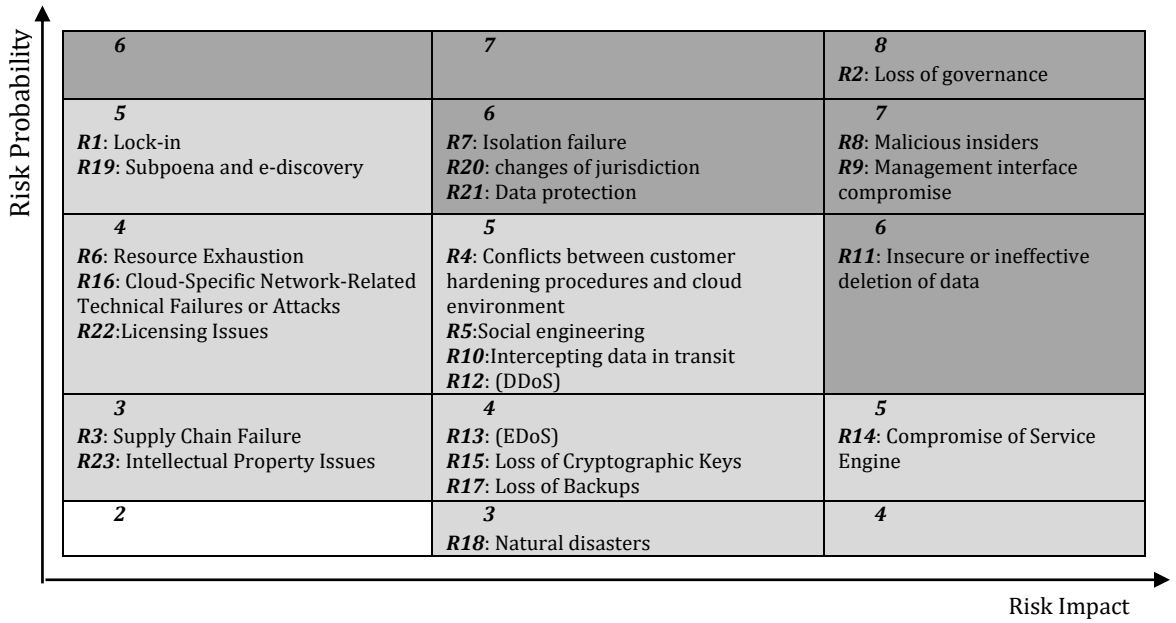

Fig. 1. The distribution of the risks probability and their impacts.

There are several types of failures that influence the reliability of cloud services. According to [11] the failures are classified as request stage failures and execution stage failures. Request stage failures affected the request before it accesses the required resource such as overflow and timeout. However, execution stage failures affected the request during its execution time such as data resource misses, computing resource missing, software failure, database failure, hardware failure, and network failure. For a cloud system to be reliable, it should be fault tolerant i.e. it should continue its operation despite any failures [12].

Availability and reliability guarantees are mentioned by some cloud providers in the Service Level Agreement (SLA). For example, in Amazon EC2 SLA, the company provides its customers a service availability of at least 99.95% which is defined as a monthly uptime percentage, and in case of service unavailability the company assigns service credits to the users with a specific percentage as a penalty [5].

3.2. Fault Tolerance Techniques in Cloud Computing

There are several techniques used to implement FTCC. According to how they deal with faults (i.e. either before or after fault occurrence.), FT techniques are classified into two main categories or policies. They are Reactive Fault Tolerance (RFT) and Proactive Fault Tolerance (PFT) as shown in Fig. 2.

RFT techniques attempt to sustain the service through recovery technique. Some of the techniques that are classified as reactive policy are given below:

- *Checkpointing/Restart*: The job restarts from the recently checked point in case of failure.
- *Replication*: Multiple copies are maintained and run on different resources for effective fault tolerance.
- *Job migration*: Tasks are migrated to other machines in case of a failure.
- *S-Guard*: It replicates failing node in the stream processing engine using rollback and recovery [13].
- *Retry***:** The concept of retrying the failed job on the cloud resource itself.
- *Task resubmission*: When a task fails, it is resubmitted either to the same or different resource [14].
- *User defined exception handling*: Treatment of a failed task is specified by the user.
- *Workflow rescue*: The workflow continues to operate despite a failure in the task [13].

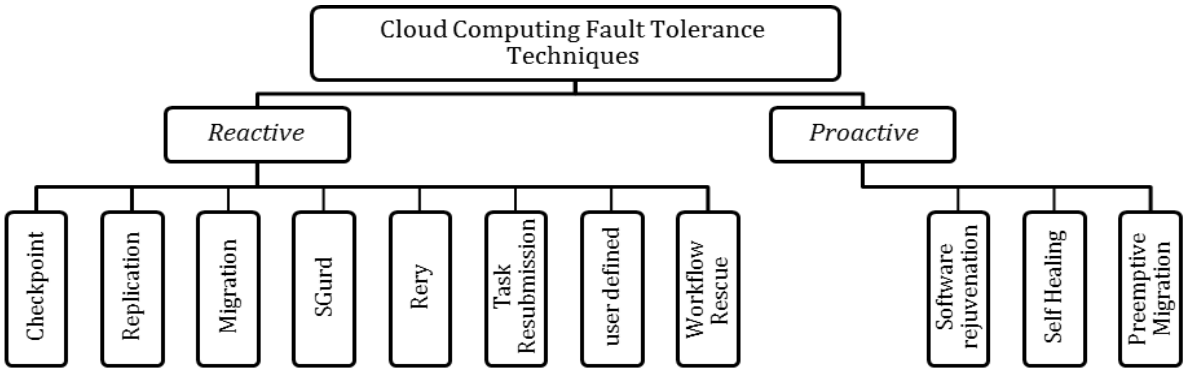

Fig. 2. Fault tolerance techniques in cloud computing.

Proactive fault tolerance techniques take some preventative measures such as to avoid any failures in the application in future [15]. Some of the techniques used are as follows:

- *Software Rejuvenation:* Restart the system with a clean state of the software [16].
- *Self-Healing:* It tolerates failure of application instances running on different Virtual Machines (VM) [17].
- *Preemptive Migration:* The application is monitored, analyzed and then preventive measures are taken.

FTCC services are still under development and studies. Moreover, new techniques emerge in the cloud from time to time. A cloud service Failure-as-a-Service (FaaS) is proposed in [18] here failure drills are run from time to time rather than waiting for unexpected failures to happen; when a drill finds a problem, recovery mechanism are initiated, thereby preventing an outage. The most popular FTCC techniques used are Checkpointing, Replication, and Job Migration. The details about the three techniques are described below:

Checkpointing: Checkpointing is a FT mechanism that takes snapshots of the system state and save it in a permanent storage (checkpoint). The system rollbacks to that state if a fault is detected by the system, instead of restarting from the beginning. Many papers have enhanced checkpoint mechanism using different algorithms. The work in [19], [20] optimized the number of checkpoints by using formulas to make it faster and reduce storage. There are several advantages of using checkpointing technique due to its low cost and high performance. However, overhead is the main disadvantage of using checkpointing.

Replication: Replication is copying all files to another storage device; the storage capacity is not an obstacle in order to improve system availability. In [21], [22] the ideal number of replica is derived and the load balance is achieved using replication. The advantages of replication technique include increased parallelism (i.e. faster query execution), higher performance, and increased speed in processing. The drawbacks of replication technique include increased overhead and cost of replication.

Job Migration: Migration is the replacement of the running VM in another VM across distinct physical hosts which separates hardware and software to make management easier. Several papers attempted to determine the time to perform VM migration. The study in [23] proposed a new method to predict the migration performance and energy cost. Advantages of job migration include easy management, load balancing, and service availability during migration. The disadvantages of job migration include cost and high overhead. Moreover, there is a possibility that the whole VM is malicious [24].

3.3. Fault Tolerance Models in Cloud Computing (FTMCC)

We categorize FTMCC into three groups according to the FT techniques discussed earlier: checkpoint based models, replication based models, and models based on multiple techniques.

3.3.1. Replication based fault tolerance: Cloud computing environments mostly have predefined hardware redundanc based FT, as the response time is a significant parameter Replication based FTCC vary in handling faults. For example, a passive replication model is capable of only tolerating crash faults while

active replication model can tolerate byzantine faults [7].

An efficient replication scheme is proposed in [25]. It transparently tolerates crash failures and offers high availability, high performance, generality, transparency, and seamless failure recovery. One of the main drawbacks of this model is the latency, as the network buffering causes performance overhead it requires additional hardware. It is an efficient replication based model but significantly introduces network delay. Thus, it is not suited for applications that are sensitive to network delay or latency. This drawback is overcome in [26] by largely reducing the external network buffering that caused the network latency. A middleware called Niagara is proposed in [27] that offers high availability and low latency. Shadow Replication is proposed in [28] to ensure successful job completion.

Byzantine Fault Tolerance (BFT) models are replication based models that are used to tolerate byzantine failures. A byzantine fault tolerance framework for voluntary-resource cloud computing is presented in [12] that tolerate faults such as: crash, arbitrary, and behaviours. The framework follows the same message algorithm that used in Practical Byzantine Fault Tolerance (PBFT) but they are different in the number of replicas. High scalability and less overhead are the advantages of using Zyzzyva [14]. A Hybrid Quorum (HQ) protocol for BFT is presented in [29]. It switches between BFT and quorum-based protocol -i.e. Query/Update (Q/U) - according to the failure occurrence.

A Virtualization and Fault Tolerance (VFT) model is represented in [30] to reduce the service time. New replication is created [31] based on selecting the finishing time of applications and dynamically generating the number of replicas. Adaptive Fault Tolerance model in Real time Cloud Computing (AFTRC) is proposed [32]. AFTRC offers both backward and forward recovery.

3.3.2. Checkpoint based fault tolerance: Checkpointing consists of three main types: coordinated checkpointing, uncoordinated checkpointing, and Communication Induced Checkpointing (CIC) [33]. CIC is an equal cost checkpointing scheme with varying checkpoint interval [4].

Checkpoint based fault tolerance for cloud computing is proposed in [34] this model uses Another Union File System (AUFS) in order to distinguish read-only properties from read and write parts in VM image. Another model with a union file system is presented in [35] that uses time storing VM checkpoints. A multi level diskless checkpointing is presented in [36].

3.3.3. Models based on multiple techniques:

Candy: A component based availability-modeling frame Work [37]. *Vega*-warden: A uniform user management system for cloud application [38]. *FT*-Cloud: A component ranking framework for fault tolerant cloud application [39]. *Magi*-Cube: A high reliability and low redundancy storage [40]. *Fault Tolerance Manager* (*FTM*) is a technique that provides FT as a service on demand for user applications [41].

4. Comparative Evaluation

For better understanding, we compare FTCC techniques and FTMCC separately. A fault tolerance model is an enhancement of a fault tolerance technique or a mix of techniques aiming to achieve better fault tolerance mechanism.

4.1. Comparing Fault Tolerance Techniques

A comparison among several FTCC used in the cloud is illustrated in Table 1. The parameters considered are policy, tool used, overhead, cost, performance and failures. *Tools used*: are the implementation techniques such as HAProxy, ASSURE, SHelp, Hadoop, Eucalyptus and AmazonEC2 [17], [35], [42]. *Overhead*: is the complexity and computation needed to achieve reliability [25], [41], [43]. *Cost*: measure of the extra resources used, software or hardware to satisfy dependability [44]. *Performance*: is a measure of the system efficiency (response time) [36]. *Detected failures*: are the types of failures that can be detected by a specific technique [5].

Table 1. Comparison between FTCC Tecniques

Techniques	Policy [34]	Tools [5]	Overhead	Cost	Performance	Failures [5]
Checkpoint	Reactive	S Help ASSURE	Average	High [45]	High [45]	Application/Host/ Network
Replication	Reactive	HAProxy Hadoop AmazonEC2	Low	Huge [36]	High [37]	Process/Node/Application
Migration	Reactive proactive	HAProxy Hadoop	Low	Varied [46]	Average [46]	Process/Node/Application
Retry	Reactive	ASSURE	High	None [4]	Average [4]	Host/ Network
SGuard [4]	Reactive	Hadoop	High	Low	Average	Node/ Application
Task Resubmission	Reactive	Amazon EC2	High	Low [13]	Average [14]	Node/ Application
Self healing[37]	Proactive	HAProxy ASSURE	Average	High [36]	High [37]	Process/Node/Application
Software rejuvenation[37]	Proactive	Eucalyptus Amazon EC2	High	High [36]	High [37]	Process/ Application[37]
Rescue workflow	Reactive	Hadoop	Low	Low [4]	Average [4]	Node/ Application

Checkpoint technique, for example, is a reactive technique that uses implementation tools such as SHelp and ASSURE. Checkpoint overhead is considered to be average since it depends on the number of checkpoints for the system and the rollback time. Checkpoint cost is considered to be high because of the memory used and storage space to save the snapshots. Checkpoint performance is high because the system can be roll backed to the nearest free fault state within a reasonable time. Three types of redundancies are required for checkpoint: storage media (hardware), snapshot application (software), and time redundancy (rollback and snapshot time). Finally the checkpoint technique can detect application, host and network failures.

4.2. Comparing Fault Tolerant Models

A comparison among most popular fault tolerance models used in cloud computing is shown in Table 2. The parameter considered are Performance: Used to evaluate the efficiency of the model. At a reasonable cost we can achieve higher performance e.g. decrease in response time while ensuring acceptable delay. *Response Time*: The time an algorithm takes to respond to a failure. *Scalability*: The ability to function effectively as the number of nodes increases. *Throughput*: The number of completed tasks. *Reliability*: The ability of the system to perform the required function in a consistent manner without any degradation. *Availability*: Ability of the system to perform its designated job, whenever required. *Usability*: The system is effective to be used by users. *Overhead Associated*: The consumed storage, time, or any resources required in completing a process.

Shadow replication, for example, is a novel fault tolerance model for cloud computing environment; it decreases failures and reduces energy consumption. The results show that shadow replication assures significant energy reduction and is preferable for compute intensive execution algorithms, where approximately up to 30% increased profit can be guaranteed due to reduction in the energy consumption. This assures both high usability and low cost, but the main drawback is the significant overhead in terms of time under shadow replication model. Processes that finish early may have to wait for other processes that are delayed by failures thus consuming static power.

Table 2. Fault Tolerance Models in Cloud Computing Comparison (H=high, L=low, A=average)

Model Name	Proactive (P) Reactive (R)/ Adaptive(A)	Response time	Scalability	Reliability	Availability	Usability	Overhead	Cost
Shadow [29]	R/A	L	L	H	A	A	A	H
AFTRC [32]	P/A	A	H	H	H	H	A	A
Remus [25]	P/R/A	H	H	H	H	L	A	A
Niagara [27]	R	A	A	H	H	L	A	L
BFT Cloud [12]	R	H	A	H	H	H	H	H
HQ-BFT [27]	R	H	H	H	H	A	H	H
Zyzzyva [15]	R	H	A	H	H	H	A	A
Magic-cube [40]	R/A	A	H	H	A	H	L	L
Checkpoint-replay Scheme [47]	R	A	H	H	H	H	L	L
Autonomic Approach [48]	P	A	A	H	H	H	A	A
Candy [37]	P/A	H	A	A	A	H	H	H

5. Conclusion

We studied and surveyed most of the available fault tolerance techniques and models that are applied in cloud computing service. Accordingly, the characteristics, advantages, and disadvantages are highlighted and tabled. Cloud providers can choose the most suitable technique or model that satisfies their requirements. Fault tolerance in cloud computing is still an active research area therefore; this work can be a reference for researchers and developers in this field. As a part of future work, we plan to simulate and evaluate the performance of the analyzed methods.

References

[1] Narde, R., Mohanish, S.,& Rajendra, R. (2013). *Fault Tolerance and Reliability in Cloud Computing.*

[2] Bort, J. (May 2016). Salesforce went down for a whole day. *Business Insider.* Retrieved from http://www.businessinsider.com/salesforce-outage-is-an-internet-meme-2016-5.

[3] CRN staff. (December 2015). The 10 biggest cloud outages of 2015 (sofar). *CRN.* Retrieved from http://www.crn.com/slide-shows/cloud/300079195/the-10-biggest-cloud-outages-of-2015.htm

[4] Bodík, P., Menache, I., Chowdhury, M., Mani, P., Maltz, D. A., &S toica, I. (August 2012). Surviving failures in bandwidth-constrained datacenters. *Proceedings of the ACM SIGCOMM 2012 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication* (pp. 431-442).

[5] Tsidulko, J. (December 2014). The 10 biggest cloud outages of 2014. *CRN. The Channel Company.* Retrieved April 5th 2014. from http://www.crn.com/slide-shows/cloud/300075204/the-10-biggest-cloud-outages-of-2014.htm?itc=refresh

[6] Avizienis, A. (Dec. 1985). The N-version approach to fault-tolerant software. *Software Engineering, IEEE Transactions, 11(12),* 1491-1501.

[7] Jhawar, R., & Vincenzo P. (2013). Fault tolerance and resilience in cloud computing environments. *Computer and Information Security Handbook,* 125-141.

[8] Kaur, J., & Kinger, S. (2014). Analysis of different techniques used for fault tolerance. *(IJCSIT) International Journal of Computer Science and Information Technologies, 5(3),* 4086-4090.

[9] Memishi, B., Shadi, I., María, S. P., & Gabriel, A. (2016). Fault tolerance in MAPREDUCE: A survey. *Resource Management for Big Data Platforms*, 205-240. Springer International Publishing.

[10] European Network and Information Security Agency. (2009). *Cloud Computing: Benefits, Risks and Recommendations for Information Security.* ENISA.

[11] Dai, Y. S., Yang, B., Dongarra, J., & Zhang, G. (November 2009). Cloud service reliability: Modeling and analysis. *Proceedings of 15th IEEE Pacific Rim International Symposium on Dependable Computing.*

[12] Zhang, Y., Zheng, Z., & Lyu, M. R. (July 2011). BFT cloud: A byzantine fault tolerance framework for voluntary-resource cloud computing. *Proceedings of 2011 IEEE International Conference on Cloud Computing (CLOUD)* (pp. 444-451). IEEE.

[13] Sindrilaru, E., Costan, A., & Cristea, V. (2010). Fault tolerance and recovery in grid workflow management systems. *Proceedings of 2010 International Conference on Complex, Intelligent and Software Intensive Systems (CISIS)* (pp. 475/480).

[14] Qureshi, K., Khan, F. G., Manuel, P., & Nazir, B. (2011). A hybrid fault tolerance technique in grid computing system. *The Journal of Supercomputing, 56(1),* 106-128.

[15] Kotla, R., Alvisi, L., Dahlin, M., Clement, A., & Wong, E. (2009). Zyzzyva: Speculative byzantine fault tolerance. *ACM Transactions on Computer Systems (TOCS), 27(4),* 7.

[16] Araujo, J., Matos, R., Maciel, P., Vieira, F., Matias, R. & Trivedi, K. S. (2011). Software rejuvenation in eucalyptus cloud computing infrastructure: a method based on time series forecasting and multiple thresholds. *2011 IEEE Third International Workshop Software Aging and Rejuvenation (WoSAR),* 38-43.

[17] Hasan, T., Imran, A., & Sakib, K. (2014). A case-based framework for self-healing paralysed components in Distributed Software applications. *Proceedings of 8th International Conference on Software, Knowledge, Information Management and Applications (SKIMA)* (pp. 1-7), IEEE.

[18] Gunawi, H. S., Do, T., Hellerstein, J. M., Stoica, I., Borthakur, D., & Robbins, J. (2011). Failure as a service (faas): A cloud service for large-scale. *Online Failure Drills, vol. 3.* Berkeley: University of California.

[19] Nicolae, B., & Cappello, F. (2011). BlobCR: Efficient checkpoint-restart for HPC applications on IaaS clouds using virtual disk image snapshots. *Proceedings of SC'11: The 24th International Conference for High Performance Computing, Networking, Storage and Analysis* (pp. 34: 1-34: 12).

[20] Malathy, G., Somasundaram, R., & Duraiswamy, K (2012). Checkpoint-based fault identification in cloud computing tasks. *International Journal of Engineering Research and Technology, 1(8).*

[21] Wei, Q. S., Veeravalli, B., Gong, B. Z., Zeng, L. F., & Feng, D. (2010). CDRM: A cost-effective dynamic replication management scheme for cloud storage cluster. *2010 IEEE International Conference on Cluster Computing (CLUSTER)* (pp. 188,196).

[22] Jhawar, R., Piuri, V., & Santambrogio, M. (2012). A comprehensive conceptual system-level approach to fault tolerance in Cloud Computing. *2012 IEEE International Systems Conference (SysCon)* (pp.1,5, 19-22).

[23] Song, F., & Xu, C. Z., (2006). Stochasticmodeling and analysis of hybrid mobility in reconfigurable distributed virtual machines. *Journal of Parallel and Distributed Computing, 66(11),* 1442-1454.

[24] Verma, A., Ahuja, P., & Neogi, A. (2008). PMAPPER: Power and migration cost aware application placement in virtualized systems. *Middleware,* 243-264. Springer Berlin Heidelberg.

[25] Cully, B., *et al.* (2008). Remus: High availability via asynchronous virtual machine replication. *Proceedings of the 5th USENIX Symposium on Networked Systems Design and Implementation.*

[26] Tamura, Y., *et al.* (2008). Kemari: Virtual machine synchronization for fault tolerance. *Proceedings of USENIX Annu. Tech. Conf. (Poster Session).*

[27] Imran, A., Gias, A. U., Rahman, R., Seal, A., Rahman, T., Ishraque, F., & Sakib, K. (2013). Cloud-niagara: A high availability and low overhead fault tolerance middleware for the cloud. *Proceedings of 16th International Conference Computer and Information Technology (ICCIT)* (271-276).

[28] Cui, X., Mills, B., Znati, T., & Melhem, R. (2014). Shadow replication: An energy-aware, fault-tolerant

computational model for green cloud computing. *Energies, 7(8)*, 5151-5176.

[29] Cowling, J., Myers, D., Liskov, B., Rodrigues, R., & Shrira, L. (November2006). HQ replication: A hybrid quorum protocol for Byzantine fault tolerance. *Proceedings of the 7th Symposium on Operating Systems Design and Implementation* (pp. 177-190). USENIX Association.

[30] Lakshmi, S. S. (2015). Fault tolerance in cloud computing. *Proceedings of National Conference on RTICCN-2015 at CGC-COE, Landran, Mohali(Punjab).*

[31] Torki, A. (Dec. 2014). A replication-based and fault tolerant Allocation algorithm for cloud computing. *International Journal of Computer Science Engineering and Technology(IJCSET).*

[32] Malik, S., & Huet, F. (July 2011). Adaptive fault tolerance in real time cloud computing. *Proceedings of 2011 IEEE World Congress on Services (SERVICES)* (pp.280, 287).

[33] Kumar, S., & Kumar, P. (2010). Hierarchical Non-blocking coordinated CHECKPOINTING algorithms for mobile distributed computing. *International Journal of Computer Science and Security (IJCSS), 3(6).*

[34] Rejinpaul, N. R., & Visuwasam, L. M. M. (2012). Checkpoint-based intelligent fault tolerance for cloud service providers. *Int'l. Journal of Computers & Distributed Systems, 2(1),* 59-64.

[35] Vallee, G., Naughton, T., Ong, H., & Scott, S. (October 2006). Checkpoint/restart of virtual machines based on xen. *High Availability and Performance Computing Workshop (HAPCW 2006).*

[36] Hakkarinen, D., & Chen, Z. (2013). Multilevel diskless CHECKPOINTING. *Computers, IEEE Transactions on, 62(4),* 772-783.

[37] Machida, F., Andrade, E., Dong, S. K. & Trivedi, K. S. (October 2011). Candy: Component-based availability Modeling framework for cloud service management using SysML. *Proceedings of 2011 30th IEEE Symposium on Reliable Distributed Systems (SRDS)* (pp. 209,218).

[38] Lin, J. Lu, X. Y., Yu, L., Zou, Y. Q & Zha, L. (July 2010). VegaWarden: A uniform user management system for cloud applications. *Proceedings of 2010 IEEE Fifth International Conference on Networking, Architecture and Storage (NAS)* (pp.457,464).

[39] Patra, P. K., Harshpreet, S., & Gurpreet, S. (2013). Fault tolerance techniques and comparative implementation in cloud computing. *International Journal of Computer Applications*, *64(14).*

[40] Feng, Q., Han, J., Gao, Y., & Meng, D. (June 2012). Magicube: High reliability and low redundancy Storage architecture for cloud computing. *Proceedings of 2012 IEEE 7th International Conference Networking, Architecture and Storage (NAS)* (pp. 89–93).

[41] Jhawar, R., Vincenzo, P., & Marco, S. (2013). Fault tolerance management in cloud computing: A system-level perspective. *IEEE Systems Journal*, *7(2),* 288-297.

[42] Araujo, J., Matos, R., Maciel, P., Vieira, F., Matias, R. & Trivedi, K. S. (2011). Software rejuvenation in eucalyptus cloud computing infrastructure: A method based on time series forecasting and multiple thresholds. *2011 IEEE Third International Workshop Software Aging and Rejuvenation (WoSAR),* 38-43.

[43] Patra, P. K., Singh, H., & Singh, G. (2013). Fault tolerance techniques and comparative implementation in cloud computing. *International Journal of Computer Applications, 64(14),* 1-6.

[44] Jhawar, R., Piuri, V., & Santambrogio, M. (2012). Fault tolerance management in cloud computing: A system-level perspective. *Systems Journal, IEEE, 7(2),* 288-297.

[45] Goiri, Í., Julia, F., Guitart, J., & Torres, J. (April 2010). Checkpoint-based fault-tolerant infrastructure for virtualized service providers. *Proceedings of 2010 IEEE Network Operations and Management Symposium (NOMS) (pp. 455-462).* IEEE.

[46] Myint, J., & Naing, T. T. (2011). Management of data replication for PC cluster-based cloud storage system. *International Journal on Cloud Computing: Services and Architecture(IJCCSA), 1(3)*.

[47] Mohammed, B., Mariam, K.,, Irfan-Ullah, A., & Kabiru, M. M. (2016). Optimising fault tolerance in real-time cloud computing IAAS environment. *Proceedings of 2016 IEEE 4th International Conference*

on Future Internet of Things and Cloud (FiCloud) (pp. 363-370). IEEE.

[48] Garg, A., & Sachin, B. (2015). An autonomic approach for fault tolerance using scaling, replication and monitoring in cloud computing. *2015 IEEE 3rd International Conference on MOOCs, Innovation and Technology in Education (MITE)* (pp. 129-134). IEEE.

Mohammad Alshayeji received his B.Sc. in computer engineering from the University of Miami and his M.Sc. in computer science from the University of Central Florida. He received his Ph.D. in computer science from University of Southern California. Currently he is working as assistant professor in Computer Engineering Department, College of Computing Sciences and Engineering, Kuwait University. Dr. Alshayeji has several publications in different international journals and conferences. His research interests include multimedia, image processing, cloud computing, security and distributed systems.

Mohammad Al-Rousan received his BSc in computer engineering from King Saud University, Saudi Arabia, in 1986. He received his M.S. in electrical and computer engineering from University of Missouri-Columbia, MI, USA in 1992. In 1996, he was awarded the Ph.D in electrical and computer engineering from Brigham Young University, UT, USA. Currently Al-Rousan is a full professor at Faculty of Computer and Information Technology, Jordan University of Science & Technology, Jordan. His technical interests include AI systems, image processing, and Nano technology

Eman Yousef received her BSc in computer science from Faculty of Science, Kuwait University, Kuwait, in 2011. She is preparing her M.S. in computer engineering from the College of Computing Sciences and Engineering, Kuwait University, Kuwait. Her technical interests include cloud computing, operating system and spatial databases.

Hanem Ellethy received her BSc in electronics and communication engineering from Faculty of Engineering, Mansoura University, Egypt, in 2005. She received her M.S. in computer engineering from the College of Computing Sciences and Engineering, Kuwait University, Kuwait, in 2016. Her technical interests include AI systems, Image processing, cloud computing and networking.