Open Access

Open Journal of Cloud Computing (OJCC)
Volume 1, Issue 2, 2014

www.ronpub.com/journals/ojcc
ISSN 2199-1987

Evaluation of Node Failures in Cloud Computing Using Empirical Data

Abdulelah Alwabel, Robert Walters, Gary Wills

Electronics and Computer Science School, University of Southampton, Southampton, SO17 1BJ, UK,
{aa1a10, gbw, rjw1}@ ecs.soton.ac.uk

ABSTRACT

Cloud has emerged as a new computing paradigm that promises to move into computing-as-utility era. Desktop Cloud is a new type of Cloud computing introduced to further achieve this ambition with an aim to reduce costs. It merges two computing models: Cloud computing and volunteer computing. The aim of Desktop Cloud is to provide Cloud services out of infrastructure that is not made for this purpose, like PCs and laptops. Such computing resources lead to a high level of volatility as a result of the fact that they can leave without prior knowledge. This paper studies the impact of node failures using evaluation metrics based on real data collected from public archive to simulate failure events in the infrastructure of a Desktop Cloud. The contribution of this paper is: (i) analysing the failure events, (ii) proposing metrics to evaluate Desktop Clouds, and (iii) evaluating several VM allocation mechanisms in the presence of node failures.

TYPE OF PAPER AND KEYWORDS

Short communication: *Cloud computing, Desktop Cloud, Node Failures, Evaluation Metrics*

1 INTRODUCTION

Desktop Clouds represent a new direction of providing Cloud services based on non-dedicated resources. The resources can be any form of computing devices such as PCs, laptops, etc. The new direction attempts to combine two computing models, namely Cloud computing and Volunteer computing, in order to form a Cloud that provides services for less or no cost, to the consumer. Throughout this paper, Traditional Cloud refers to a Cloud that relies on dedicated resources to provide services, whereas Desktop Cloud refers to a Cloud that relies on non-dedicated resources [1]. Amazon Cloud, for instance, is a Traditional Cloud.

Desktop Clouds are built on top of computing resources that are prone to failure at any time without prior knowledge. Such volatile infrastructure can have negative impact upon the running and output of Desktop Clouds. The contribution of this paper is threefold. Firstly, the paper introduces Desktop Cloud systems as being a new type of Cloud computing, and compares it with related systems, Traditional Clouds and Desktop Grids, in order to clarify Desktop Clouds further. Secondly, the work proposes three metrics that can be used to evaluate VM mechanisms employed by Cloud middleware platform. Finally, the paper simulates a Desktop Cloud using empirical data of node failures collected online from SETI@home system. The impact of node failures on Desktop Clouds are evaluated using the proposed metrics.

The remaining of this paper is organised as follows. First, the paper discusses Desktop Cloud computing model by comparing it with related systems. Several research challenges in Desktop Clouds are discussed. A

further attention is given to the failure node issue in Desktop Cloud. Node failures are analysed using empirical data collected from public archive which provides log files of SETI@home system. Several VM allocation mechanisms are experimentally evaluated in the presence of node failures. Section 4 reports and discusses results of the experiments. Finally, the paper concludes with our insights in the future direction of research in Desktop Clouds.

2 DESKTOP CLOUDS

The success of Desktop Grids motivates the idea of using idle resources to build Desktop Clouds. Note that the term Desktop is taken from Desktop Grids because the infrastructure of both of Desktop Clouds and Desktop Grids are made of Desktop PCs and laptops, etc. Similarly, the term Cloud comes from Cloud computing since the services in Desktop Clouds is provided on the Cloud business basis. There are several synonyms for Desktop Cloud used on the literature, such as Ad-hoc Cloud [2], Volunteer Cloud [3], Community Cloud [4] and Non-Dedicated Cloud [5].

Ad-hoc Cloud is the idea of harvesting distributed resources within an organisation to form a Cloud [2]. Nebula [6][7] is a project aiming to exploit distributed resources in order to create a volunteer Cloud which offers services free of charge. Cloud@home [8][9] is a project representing the @home philosophy in Cloud computing. The goal of Cloud@home is to form a new model of Cloud computing contributed by individual users over the Internet. In addition to that, Cern (the European Organization for Nuclear Research) [10] has announced an initiative to move their Desktop Grid project, which is called LHC@home, toward the Cloud. It is suggested that non-dedicated resources can be exploited by Cloud providers in case their local infrastructure cannot meet requests by consumers at peak times [5].

An overview of the architecture of Desktop Clouds is depicted in Figure 1. The architecture is consisted of several layers. The users contact the service layer in order to submit their demands. The physical layer is responsible of managing physical nodes that are aggregated in the resource layer. The virtual layer plays a curtail role in terms of isolating client requests from the physical nodes via virtualisation. Users are assigned virtual machines that are located in physical machines. Physical machines can be connected by LAN or WLAN.

As a scenario of building a private Cloud, a group of universities wishes to benefit from its computing

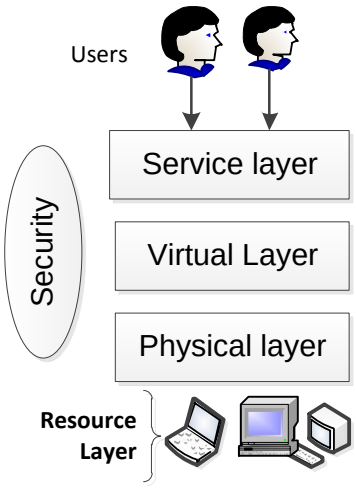

Figure 1: Architecture of Desktop Clouds

resources to form a Cloud. The resources range from PCs to servers, etc., each of them is called a Cloud node. A node can join the Cloud when it becomes idle. This scenario is motivated by Condor [11]. Users in Desktop Cloud submit their request to acquire services with the requirements stated in the service level agreement between a client and the Cloud interface. The requests are processed in the virtualisation layer on top of Cloud physical nodes. The virtualisation isolates the guest operating system from the host physical machine. The isolation improves security and prevents unauthorised access between two parties.

Another scenario that can be considered is a universal Desktop Cloud, which allows people to contribute their own computing resources to be used by Cloud clients [12]. This example can be considered as public Desktop Cloud. The people are asked to contribute their machines in order to form a Desktop Cloud. People can be stimulated to participate in Desktop Cloud to serve science within research communities.

Table 1: Desktop Cloud vs. Traditional Cloud

Feature	Desktop Cloud	Traditional Cloud
Elasticity	√	√
Virtualisation	√	√
Idle Resources	√	X
Ease of Use	√	√

2.1 Desktop Clouds vs. Related Systems

This subsection explains further the model of Desktop Clouds by comparing it with related areas: Traditional Clouds and Desktop Grids. There are some differences between Desktop Cloud and Traditional Clouds as shown in Table 1. The meaning of Elasticity is that consumers can acquire computing services and scale them up or down according to their needs. Both Traditional Cloud and Desktop Cloud rely heavily on virtualisation. Furthermore, the infrastructure of Desktop Cloud is made of resources that are not dedicated as nodes in the Cloud infrastructure. In the contrary, the infrastructure of Traditional Cloud is made of a huge number of dedicated computing resources. The ease of use principle means that users can use a specific service without making a lot of changes to their work. Both Traditional Clouds and Desktop Clouds let their users harness services without making significant changes to their code.

Table 2: Desktop Cloud vs. Desktop Grid

Feature	Desktop Cloud	Desktop Grid
Elasticity	√	X
Virtualisation	√	X
Idle Resources	√	√
Ease of Use	√	X

Desktop Clouds may be confused with Desktop Grids because they both rely on similar infrastructure. Table 2 shows a comparison between Desktop Cloud and Desktop Grid. Both computing models serve the same goal that aims at employing idle computing resources which can be denoted by the public or limited only to resources within an organisation. Virtualisation is not employed in Desktop Grids. People who wish to contribute their computing machines need to install a specific software in order to join a Desktop Grid. Furthermore, users in Desktop Grids need to know in depth about the structure of a Desktop Grid system. Such notation violates the feature of elasticity [13].

2.2 Research Challenges

This section discusses several research issues that need further attention. These research challenges are security, resource management and node failures. Some of these challenges are inherited from Cloud computing, while others are driven by the nature of the employed resources being highly volatile.

Security is one of the major concerns that prevent organisations from moving to the cloud [14]. Ristenpart et al. [15] show that an attacker can uncover the actual location of a particular virtual machine (VM). Then, a cross-VM side channel attack can reveal critical information about the targeted VM by placing a malicious VM on the same physical machine. More worries arise in Desktop Clouds where both consumers and contributors are from the public. Therefore, security can be a major issue in this context. In addition to the previous threats presented in the cloud, both consumers and contributors take on risk themselves when they join a Desktop Cloud. A contributor can put his own data at risk by allowing access to a virtual image located in his machine. Likewise, consumers are vulnerable to malicious contributors. Nodes in Desktop Cloud are more likely to be vulnerable to outside attacks due to weaknesses in local antivirus software and firewalls.

Virtualisation can be vital in order to isolate the host completely from guest operating systems, and thus preventing any unwanted access from either party. Trust mechanisms can be employed in this matter. For example, a Desktop Cloud can maintain a behaviour table which contains information about both consumers and contributors. The table can be used to decide which parties are trustworthy enough to join the cloud. Furthermore, Desktop Clouds should rely on autonomous mechanisms, such as sandbox or certification, in order to prevent various attacks from participants [16].

Resources in Desktop Clouds are highly heterogeneous, and managing them therefore can be considered problematic. Virtualisation plays a key role in Desktop Clouds because it virtualises contributed resources and delivers them to users as VMs. Desktop Clouds face a challenge of developing a resource allocation mechanism that is able to: a) manage non-dedicated, heterogeneous resources, b) deliver a virtualized machine to upper level in Desktop Clouds and c) work closely with users' tasks in order to find most suitable nodes for each request.

It has been pointed out that lacking central management in Desktop Clouds cause a major issue in terms of reliability and state maintenance in case of failures [17]. The infrastructure of Desktop Cloud is consisted of nodes that are highly volatile. Therefore, fault recovery mechanisms are crucial in order to improve reliability in this environment [18]. In addition, Desktop Clouds require means to interact with other clouds for data migration or to gain extra computing resources [8].

Desktop Clouds are expected to offer services at a low level of reliability and availability due to the fact that they depend on unreliable volunteered resources, which can join or leave the cloud without prior knowledge for various reasons [5]. Availability of individual nodes is considered a primary issue in Desktop Clouds [18]. For example, it is estimated that

resource unavailability can reach up to 50% in volunteer projects [19]. Availability of each individual node can affect the service quality. Andrzejak et al. [20] propose a technique to predict the availability of a group of high volatility resources.

2.3 Failure Study

We study, in this subsection, further the failure of nodes in traditional clouds because it can help in assessing the impact of failures on Desktop Clouds. However, to the best of our knowledge, there is no actual Desktop Cloud system available to analyze failures over there. As a result, we study SETI@home Desktop Grid instead, because a Desktop Cloud can use the infrastructure of a Desktop Grid, as mentioned in subsection 2.1. The logs of nodes of SETI@home can be collected from the Failure Trace Archive (FTA).

The FTA is a repository, which is available online, containing traces of several distributed and parallel systems [21]. The archive includes a wide range of traces for several distributed systems, including Grid computing, High Performance Computing, Desktop Grid and peer-to-peer systems. Each system's archive contains timestamp events that are recorded periodically for each node of the system. Each event refers to the state of the associated node. For example, a state of an event can be unavailable, and this means that this node is down at the given timestamp of the event. Two event states are considered node failure in this study. The event states, in the FTA, are unavailable and failure states.

The files of SETI@home were collected from the University of Notre Dame. The FTA has a large pool of resource (more than 200 thousand nodes) that have been run for one-year period in 2008/09 [22]. The nodes in SETI@home are highly heterogeneous because most of these computing nodes are denoted by the public over the Internet. A random sample of 875 nodes has been selected from SETI@home FTA for six months. We calculated the average percentage failure of SETI@home's nodes on every hour basis. Such study can help in evaluating the behaviour of VM mechanisms. The failure percentage is calculated as shown in equation 1:

$$failure(h) = \frac{\sum(failednodes)_h * 100}{totalnumberofnodes} \quad (1)$$

Although the archive provides traces of the behaviour of nodes, it needs some analysis to calculate the failure events. Several papers in the literature studied the failure events in the FTA archive such as [23][24][25]. The literature shows that the focus is on the availability behaviour of nodes. The availability means that the time of a specific machine remains available. Studying the behaviour of machines can yield in discovering statistical models of availability in desktop Grids [25]. Such statistical model can help in predicting availability of machines in order to improve source selection mechanisms as mentioned in [24]. The case in Desktop Clouds is different because the number of failures matters more than the availability time of resources. A failure of node causes that all hosted VMs in the node fail.

Table 3: SETI@home failure summary

Hours	4320
Mean	13.67 %
Median	12.47 %
Std. Deviation	5.84
Minimum	3.43 %
Maximum	76.77 %

Table 3 shows a summary of 4320 hours (6 months * 30 days * 24 = 4320 hours). On average, about 14% of nodes fail per hour. Failure in this context can involve a node leaving the system. Figure 3 shows an average hourly failure percentage in 24 hour-period for analysis of 6 months period of SETI@home nodes. The six-month period is divided into days, which makes it 180 day. The period is set to 24 hours because this is the running time set for our experiments. Hour 1 recorded the highest failure percentages at about 21.15% while Hour 9 was the lowest at about 9.7%. These figures can express that failure events in Desktop Clouds are norms rather than exceptions.

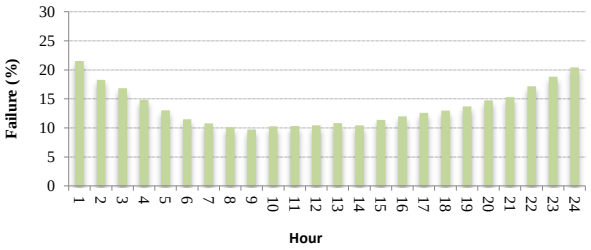

Figure 2: SETI@home Average Failure Percentage

2.4 Evaluation Metrics

The outcome of a Cloud system can be measured by different metrics according to the perspective. For example, Cloud brokers and customer are interested in the performance and cost aspects, such as response time, down time, etc. [26]. From the prospective of service providers, it is crucial to reduce the running costs. This can be achieved by employing techniques and mechanisms to reduce power consumed by the nodes in

the infrastructure level [27]. In our work, we considered three metrics that can be used to evaluate VM allocation mechanism employed in a Desktop Clouds. VM allocation mechanism is the process of allocating a VM to a PM.

The proposed metrics in this research are throughput, power consumption and availability. The throughput metric measures the number of successfully completed tasks *st* that are submitted by clients out of the total number of submitted tasks *tt* [28]. It is important to consider the throughout output in the presence of node failures. Throughput is calculated as shown in Equation 2:

$$throughput = \frac{\sum st \ * \ 100}{tt} \qquad (2)$$

Power consumption is the amount of energy *p* that consumed by each node in the infrastructure layer of a Cloud system. It is measured by Kilo Watt per hour (kWh). The metric of power consumption is given as shown in Equation 3:

$$power\ consumption = \sum_{i=0}^{n} p(node_i) \qquad (3)$$

The last metric measures that how much resource computing power is available to serve new requests. The failure of nodes affects the availability of Desktop Clouds. A question can be raised in this context of about whether the employed VM allocation mechanism can help in reducing this effect. Each available node's computing power is *avl* while the total computing power is *tot.cp*. The availability is calculated as shown in Equation 4:

$$availability = \frac{\sum available\ nodes}{tot.cp} \qquad (4)$$

3 EXPERIMENT

The experiment is conducted to study the impact of node failures on Desktop Clouds. Four VM allocation mechanisms are evaluated using DesktopCloudSim simulation tool. The tool is used to simulate public Desktop Clouds based on SETI@home FTA.VM allocation mechanism is the process of allocation VMs by Cloud's users to physical machines (PMs). It also involves the process of migrating VMs between PMs during run time.

The tested mechanisms are FCES (first come first serve), Greedy, RoundRobin and Random mechanisms. The FCFS mechanism allocates a VM to the first available PM that can accommodate it [29]. Greedy mechanism allocates as many VMs as possible to the same PM to improve utilisation of resources [30]. The RoundRobin mechanism allocates to each PM the same number of VMs. The Random mechanism allocates VMs randomly to PMs [31]. These mechanisms are choosed because they are implemented in open source Cloud management platforms, such as Eucalyptus [20], OpenNebula [21] and Nimbus [22].

3.1 DesktopCloudSim

DesktopCloudSim is an extension tool for CloudSim developed to simulate failure events happening in the infrastructure level in Cloud computing. CloudSim is a Java-based discrete event simulation toolkit designed to simulate Cloud computing [32]. Although CloudSim is widely used by researchers to study various issues in Cloud computing, the tool lacks some important features. Performance variations of VMs are not simulated in CloudSim when they process tasks [33]. CloudSim does not simulate service failures in the service layer [34]. Furthermore, CloudSim lacks the ability to simulate dynamic interaction of nodes in the infrastructure level. CloudSim allows static configuration of nodes which remain unchanged during run time. Finally, node failures are not included in CloudSim tool. DesktopCloudSim enables the simulation of dynamic nodes and node failures.

3.2 Experiment Setting

The experiment is run for 180 times, each run time represents a simulation of a day running of SETI@home nodes i.e. on a daily basis. Each VM allocation mechanism is run for 180 times representing traces of 6 months collected from the FTA. The run time set to one day because the FTA provides a daily trace for SETI@home nodes as discussed in subsection 2.3.

There are two input data sets augmented to DesktopCloudSim. The first is the FTA files to simulate nodes in the infrastructure along with failure event times. The second input data set is a workload representing the tasks submitted by users to be processed in VMs. The workload was collected from the PlanetLab archive. The archive provides traces of real live applications submitted to the PlanetLab infrastructure [35]. One day workload was retrieved randomly as input data in this experiment. The workload input remains the same during all the experiment runs because the aim of this experiment is to study the impact of node failures on throughput of Desktop Clouds.

The FTA files provide the number and IDs of nodes. However, the specifications of nodes are missing from the archive. As a result, the specifications are set randomly for physical machines. The missing specifications are technical specifications, such as CPU power, RAM size and hard disk size. The number of

requested VM instances is 700 instances run for 24 hours. VM instances are classified into: *micro, small, medium and large* VM types as offered by Amazon EC2. Each VM instance receives equally a series of tasks to process for a given workload.

If a node fails, in the experiment, then all hosted VMs will be destroyed. The destruction of a VM causes all running tasks on the VM to be set as failing. The lost VM is restart again and allocated to another PM to process tasks. The simulation is run on a Mac i27 (CPU = 2.7 GHz Intel Core i5, 8 GB MHz DDR3) running OS X 10.9.4. The results were analysed using IBM SPSS Statistics v21 software.

4 RESULTS AND DISCUSSION

The experiment was conducted to compare four VM allocation mechanisms: FCFS, Greedy, RoundRobin and Random mechanisms to study the impact of node failures. The mechanisms are evaluated using metrics proposed in subsection 2.4. Each evaluation metric is analysed separately in a different section. Each section reports the results of a public Cloud (represented by SETI@home data set).

4.1 Throughput

Table 4 shows a summary of descriptive results obtained when measuring the throughput metric for each VM allocation mechanism for SETI@home Cloud. N in the table means that the number of days which is 180 days representing a six-month period. Kolmogorov-Smirnov (K-S) test of normality shows that the normality assumption was not satisfied because Greedy mechanism is statistically significantly non-normal, $P < .05$. Therefore, the non-parametric test Friedman's ANOVA was used to test which mechanism can yield better throughput. Friedman's ANOVA test confirms that throughput varies statistically significantly from one mechanism to another, $X_F^2(3) = 269.06, P < .001$. Mean, median, variance and standard deviations are reported in Table 4.

Table 4: Throughput metric results

Mechanism	N	Mean (%)	Median (%)	Var.	St. Dev.
FCFS	180	78.62	79.81	18.59	4.31
Greedy	180	78.4	78.52	14.78	3.84
RoundRobin	180	77.1	77.67	14.79	3.85
Random	180	80.24	80.82	11.79	3.43

Six Wilcoxon pairwise comparison tests were used to find out the mechanism with the highest throughput. Note that 6 tests required to compare 6 pairs of mechanisms: FCFS vs. Greedy, FCFS vs. RoundRobin, FCFS vs. Random, Greedy vs. RoundRobin, Greedy vs. Random and RoundRobin vs. Random. The level of significance was set to 0.008 using Bonferroni correction method [36] because there were 6 post-hoc tests required ($.05/6 \approx .008$). The tests showed that there are statistically significant differences between all pairs except between the FCFS vs. Greedy mechanisms because the test showed that the difference was not statistically significant, as Table 5 shows. However, this does not affect the overall finding: The Random mechanism is the best with median of about 81%. It is worth mentioning that the difference between all mechanisms is very little by about 3% only.

Table 5: Pairwise comparisons using Wilcoxon tests for throughput metric

Pairwise mechanism comparison	Wilcoxon test
FCFS vs. Greedy	P >.008
FCFS vs. RoundRobin	P < .008
FCFS vs. Random	P < .008
Greedy vs. RoundRobin	P < .008
Greedy vs. Random	P < .008
RoundRobin vs. Random	P < .008

4.2 Power Consumption

This section evaluates the FCFS, Greedy, RoundRobin and Random mechanisms in terms of power consumed by nodes in the Cloud. Both Greedy and Random mechanisms are statistically significantly non-normally distributed, $P < .05$. Therefore Friedman's ANOVA test was applied to test if there is a statistically significant difference between the rests. The test showed that there is indeed a statistically significant difference, $X_F^2(3) = 540, P < .001$. The mean, median, variance and standard deviation is reported in Table 6.

Table 6: Power consumption metric results

Mechanism	N	Mean (kWh)	Median (kWh)	Va.	St. Dev.
FCFS	180	507	506	131.85	11.48
Greedy	180	694	696	614.7	24.79
RoundRobin	180	2217	2215	2185	46.74
Random	180	1533	1534	3263	57.12

Pairwise comparison tests were conducted to find out which mechanism is better in terms of power consumption, as in Table 7. There are statistically significant differences between the tested pairs' mechanisms. The FCFS mechanism consumes less power than other mechanisms, median = 506 kWh. In addition, Greedy mechanism comes second while the Random and RoundRobin come third and fourth respectively.

Table 7: Pairwise comparisons using Wilcoxon tests for power consumption metric

Pairwise mechanism comparison	Paired-samples T test
FCFS vs. Greedy	P < .008
FCFS vs. RoundRobin	P < .008
FCFS vs. Random	P < .008
Greedy vs. RoundRobin	P < .008
Greedy vs. Random	P < .008
RoundRobin vs. Random	P < .008

4.3 Availability

Table 8 shows mean, median, variance and standard deviations results obtained when measuring the availability metric for each VM allocation mechanism in SETI@home Cloud. Kolmogorov-Smirnov test of normality yields that the results of mechanisms are normally distributed, P > .05. Mauchly's test indicated that the assumption of sphericity had been violated $x^2(5) = 58.57, p < .05$. Therefore, the degree of freedom was corrected by Greenhouse-Geisser [36] estimates of sphericity ($\varepsilon = .82$). The test shows power consumed by Clouds' nodes was significantly affected by the employed VM allocation mechanism, $F(2.45, 438.65) = 8265.29, p < .05$.

Table 8: Availability metric results

Mechanism	N	Mean (%)	Median (%)	Va.	St. Dev.
FCFS	180	91.81	91.8	.06	.23
Greedy	180	92.59	92.6	.1	.31
RoundRobin	180	88.83	88.84	.1	.31
Random	180	88.67	88.65	.12	.34

The repeated ANOVA test showed that the availability varies significantly. Several pairwise comparisons using Paired T-test were conducted. Table 9 gives the results of these tests, which shows that there are significant differences between node availability for each VM mechanism. Therefore, it can be said that Greedy mechanism has the highest availability when it was used in a public Cloud.

Table 9: Pairwise comparisons using paired T-tests for availability metric

Pairwise mechanism comparison	Paired T-test
FCFS vs. Greedy	P < .008
FCFS vs. RoundRobin	P < .008
FCFS vs. Random	P < .008
Greedy vs. RoundRobin	P < .008
Greedy vs. Random	P < .008
RoundRobin vs. Random	P < .008

4.4 Summary

Table 10 summarises experiment findings of which VM allocation mechanism yields best results in the sight of three metrics: throughput, power consumption and availability. The Random mechanism is the best in terms of throughput. The FCFS mechanism consumes least power in a public Cloud. The Greedy mechanism was the best in terms of node availability.

Table 10: Best mechanism in each metric

Metric	VM mechanism
Throughput	Random
Power consumption	FCFS
Availability	Greedy

The Random mechanism allocates VM randomly which makes it difficult to reason why it behaves the best. However, approximate one fifth of the submitted tasks are not successfully completed. Such a figure demonstrates the importance of developing a fault-tolerant mechanism for Desktop Clouds. Both the FCFS and Greedy mechanisms are efficient in terms of power consumption because they try to improve utilisation. For Availability of resources, Greedy mechanism beats other mechanism by about 3.14%. This shows that VM allocation mechanisms play a very small roll in terms of improving availability. We can conclude that the impact of an average of node failure at about 14% on throughput is about 19% of tasks failed.

5 CONCLUSION

Desktop Cloud has approached as being an alternative solution to provide capabilities of Cloud using infrastructure that is made of normal computer PCs and laptops. Such computing resources can join the Cloud when they become idle. Desktop Cloud can be used to reduce costs of exploiting Cloud services. However, such benefits come with a negative impact of the outcome of Desktop Cloud as a result of node failures. The study of failures conducted in this paper shows that the failure of node is quite high.

This paper has evaluated the impact of node failures using three metrics: throughput, power consumption and availability. Throughput is a metric to measure the number of success rate of tasks submitted by clients. The power consumed by each node in the Cloud is measured via power consumption metric. Availability metric measures the percentage of available computing resources ready to serve new requests.

The experiment has been conducted on DesktopCloudSim, which is a simulation extension for CloudSim tool to enable simulation of node failures. Four VM allocation were evaluated: FCFS, Greedy, RoundRobin and Random mechanisms. The node failures are simulated based on traces files collected from SETI@home data. Our simulation has shown that the throughput of Desktop Clouds is greatly affected by node failures. As a result of our experiment, we recommend that there is a need to develop fault-tolerant mechanism to overcome the impact of node failures.

REFERENCES

[1] A. Alwabel, R. Walters, and G. Wills, "A view at desktop clouds", in *ESaaSA* 2014.

[2] G. Kirby, A. Dearle, A. Macdonald, and A. Fernandes, "An Approach to Ad hoc Cloud Computing", *Arxiv Prepr. arXiv1002.4738*, 2010.

[3] B. M. Segal, P. Buncic, D. G. Quintas, D. L. Gonzalez, A. Harutyunyan, J. Rantala, and D. Weir, "Building a volunteer cloud", *Memorias la ULA*, 2009.

[4] A. Marinos and G. Briscoe, "Community Cloud Computing", in *Proceedings of the 1st International Conference on Cloud Computing*, pp. 472–484, 2009.

[5] A. Andrzejak, D. Kondo, and D. P. Anderson, "Exploiting non-dedicated resources for cloud computing", *2010 IEEE Netw. Oper. Manag. Symp. - NOMS 2010*, pp. 341–348, 2010.

[6] A. Chandra and J. Weissman, "Nebulas: Using distributed voluntary resources to build clouds", the 2009 *conference on Hot topics in cloud computing*, 2009.

[7] J. B. Weissman, P. Sundarrajan, A. Gupta, M. Ryden, R. Nair, and A. Chandra, "Early experience with the distributed nebula cloud", in *Proceedings of the fourth international workshop on Data-intensive distributed computing*, pp. 17–26, 2011.

[8] V. D. Cunsolo, S. Distefano, A. Puliafito, and M. Scarpa, "Volunteer computing and desktop cloud: The cloud@ home paradigm", in *Proceedings of Eighth IEEE International Symposium on Network Computing and Applications*, pp. 134–139, 2009.

[9] V. Cunsolo and S. Distefano, "From volunteer to cloud computing: cloud@ home", in *Proceedings of the 7th ACM international conference on Computing frontiers*, pp. 103–104, 2010.

[10] A. Harutyunyan, J. Blomer, P. Buncic, I. Charalampidis, F. Grey, A. Karneyeu, D. Larsen, D. Lombraña González, J. Lisec, B. Segal, and P. Skands, "CernVM Co-Pilot: an Extensible Framework for Building Scalable Computing Infrastructures on the Cloud", *J. Phys. Conf. Ser.*, vol. 396, no. 3, p. 032054, Dec. 2012.

[11] M. J. Litzkow, M. Livny, and M. W. Mutka, "Condor-a hunter of idle workstations", in *Proceedings of 8th Int. Conf. Distrib.*, pp. 104–111, 1988.

[12] V. Cunsolo, S. Distefano, A. Puliafito, and M. Scarp, "Cloud@ home: Bridging the gap between volunteer and cloud computing", *5th Int. Conf. Emerg. Intell. Comput. Technol. Appl.*, 2009.

[13] I. Foster, Y. Zhao, I. Raicu, and S. Lu, "Cloud computing and grid computing 360-degree compared", in *Proceedings of Grid Computing Environments Workshop*, pp. 1–10, 2008.

[14] T. Dillon, C. Wu, and E. Chang, "Cloud computing: Issues and challenges", in *Proceedings of 24th IEEE International Conference on Advanced Information Networking and Applications*, pp. 27–33, 2010.

[15] T. Ristenpart, E. Tromer, S. Savage, and H. Shacham, "Hey, you, get off of my cloud: exploring information leakage in third-party compute clouds", in *Proceedings of the 16th ACM conference on Computer and communications security*, pp. 199–212, 2009.

[16] B. Q. Cao, B. Li, and Q. M. Xia, "A Service-Oriented Qos-Assured and Multi-Agent Cloud Computing Architecture", *Cloud Comput.*, pp. 644–649, 2009.

[17] P. Endo, A. de A. Palhares, N. N. Pereira, G. E. Goncalves, D. Sadok, J. Kelner, B. Melander, and J.-E. Mangs, "Resource allocation for distributed cloud: concepts and research challenges", *Network, IEEE*, vol. 25, no. August, pp. 42–46, 2011.

[18] A. Marosi, J. Kovács, and P. Kacsuk, "Towards a volunteer cloud system", *Futur. Gener. Comput. Syst.*, Mar. 2012.

[19] D. Kondo, M. Taufer, and C. Brooks, "Characterizing and evaluating desktop grids: An empirical study", *Int. Parallel Distrib. Process. Symp.*, vol. 00, no. C, 2004.

[20] A. Andrzejak, D. Kondo, and D. P. Anderson, "Ensuring collective availability in volatile resource pools via forecasting", *Manag. Large-Scale Serv.*, no. contract 34084, pp. 149–161, 2008.

[21] B. Javadi, D. Kondo, A. Iosup, and D. Epema, "The Failure Trace Archive: Enabling the comparison of failure measurements and models of distributed systems", *J. Parallel Distrib. Comput.*, vol. 73, no. 8, pp. 1208–1223, Aug. 2013.

[22] B. Javadi, D. Kondo, J. Vincent, and D. P. Anderson, "Mining for Statistical Models of Availability in Large-Scale Distributed Systems : An Empirical Study of SETI @ home," in *Proceedings of IEEE International Symposium on Modeling, Analysis & Simulation of Computer and Telecommunication Systems*, pp. 1 – 10, 2009.

[23] B. Rood and M. J. Lewis, "Multi-state grid resource availability characterization", in *Proceddings of 8th IEEE/ACM Int. Conf. Grid Comput.*, pp. 42–49, Sep. 2007.

[24] F. Nadeem, R. Prodan, and T. Fahringer, "Characterizing, Modeling and Predicting Dynamic Resource Availability in a Large Scale Multi-purpose Grid", in *Proceedings of Eighth IEEE Int. Symp. Clust. Comput. Grid*, pp. 348–357, May 2008.

[25] B. Javadi, D. Kondo, J. Vincent, and D. P. Anderson, "Discovering statistical models of availability in large distributed systems: An empirical study of SETI@ home", *Parallel Distrib. Syst. IEEE Trans.*, vol. 22, no. 11, pp. 1896–1903, 2011.

[26] V. C. Emeakaroha, M. a. S. Netto, R. N. Calheiros, I. Brandic, R. Buyya, and C. a. F. De Rose, "Towards autonomic detection of SLA violations in Cloud infrastructures", *Futur. Gener. Comput. Syst.*, vol. 28, no. 7, pp. 1017–1029, Jul. 2012.

[27] A. Beloglazov, J. Abawajy, and R. Buyya, "Energy-aware resource allocation heuristics for efficient management of data centers for Cloud computing", *Futur. Gener. Comput. Syst.*, vol. 28, no. 5, pp. 755–768, May 2012.

[28] S. K. Garg, S. Versteeg, and R. Buyya, "A framework for ranking of cloud computing services", *Futur. Gener. Comput. Syst.*, vol. 29, no. 4, pp. 1012–1023, Jun. 2013.

[29] U. Schwiegelshohn and R. Yahyapour, "Analysis of first-come-first-serve parallel job scheduling", in *Proceedings the ninth annual ACM-SIAM symposium on Discrete algorithms*, pp. 629–638, 1998.

[30] J. Cunha, P. Kacsuk, and S. Winter, "*Parallel Program Development for Cluster Computing: Methodology, Tools and Integrated Environments"*, Nova Biomedical, 2001.

[31] R. Rasmussen and M. Trick, "Round robin scheduling–a survey", *Eur. J. Oper. Res.*, pp. 617–636, 2008.

[32] R. Buyya, R. Ranjan, and R. N. Calheiros, "Modeling and simulation of scalable Cloud computing environments and the CloudSim toolkit: Challenges and opportunities", *High Perform. Comput. Simulation*, pp. 1–11, June 2009.

[33] M. Bux and U. Leser, "DynamicCloudSim: simulating heterogeneity in computational clouds", *2nd ACM SIGMOD Workshop on Scalable Workflow Execution Engines and Technologies*, 2013.

[34] W. Chen and E.Deelman, "WorkflowSim : A Toolkit for Simulating Scientific Workflows in Distributed Environments", IEEE 8th International Conference on E-Science, 2012.

[35] L. Peterson, S. Muir, T. Roscoe, and A. Klingaman, "PlanetLab Architecture : An Overview", PDN–06–031, PlanetLab, May 2006.

[36] A. Field, "Discovering statistics using SPSS", 3rd edition, SAGE Publications Ltd, 2009.

AUTHOR BIOGRAPHIES

Abdulelah Alwabel is a PhD student at the University of Southampton. His research focus is to develop fault-tolerant mechanisms for VM allocation process in Desktop Clouds. Abdulelah obtained his master degree in Computer Science from the University of Bristol in 2010.

Dr. Robert Walters is an Assistant Professor in Computer Science at the University of Southampton. His research interests include middleware, distributed computing, hypermedia and graphical formal modeling language.

Dr. Gary Wills, BEng, PhD, CEng is an Associate Professor at the University of Southampton. Gary's research projects focus on System Engineering and is underpinned by technologies such as, Secure Systems, Distributed Systems, SOAs and Cloud Computing