The Ability of Cloud Computing Performance Benchmarks to Measure Dependability

Eduardo Carvalho[1], Raul Barbosa[2] and Jorge Bernardino[1,2]
[1]*ISEC, Polytechnic of Coimbra, Rua Pedro Nunes, Coimbra, Portugal*
[2]*CISUC - Centre for Informatics and Systems of the University of Coimbra, Coimbra, Portugal*

Keywords: Cloud Benchmarking, Dependability, Fault Tolerance, Cloud Computing.

Abstract: Current benchmarks focus on evaluating performance with little efforts made to evaluate the dependability characteristics of the cloud. Cloud computing has several advantages like scalability, elasticity and cost reduction, this led companies to move their applications to the cloud. The availability of applications and consequently businesses are then dependant on the cloud's efforts to keep its services running. To guarantee reliability and trust, benchmarking dependability is a challenging task because of the cloud layered model which makes it difficult to predict the root of faults as higher layers are dependant of lower layers. By integrating dependability in benchmarks as a metric, we can evaluate how well does the cloud handle itself when faults occur, prevent those faults and check not only raw performance but also trust. In this paper, we study the following cloud benchmarks: Spec IaaS 2016, TPCx-V, YCSB, Perfkit Benchmarker, CloudBench, DS-Bench/D-Cloud, and evaluate if they are suitable for benchmarking dependability.

1 INTRODUCTION

Cloud computing offers on-demand dynamically scalable resources such as computing, storage and networking (Vazquez et al., 2014). These resources come as a service to the user and are priced per usage. The reduction in cost combined with the cloud's "infinite resources" illusion, made this paradigm very appellative to companies who wanted to grow fast without the risk of investing in superfluous resources. Benchmarking these systems to compare the services that providers offered became mandatory, before choosing what service to acquire and what provider presented the best solution. In the last years, emerged cloud benchmarks that envisaged the cloud as a paid service and benchmarked more than database performance (Abramova et al., 2014) (Neves et al., 2016). The new generations of cloud benchmarks tend to evaluate the characteristics of the cloud not only separately but also as a cooperative scheme because they affect each other (Neves and Bernardino, 2015) (Oppenheimer et al., 2002).

Cloud computing is complex and large-scale, which makes it hard to benchmark and prone to faults. When managing these systems, faults are easy to occur which may lead to errors and consequently unpredicted failures (Guan et al., 2012). A failure makes services unable to accomplish their function, since these services are being contracted, this can lead to loss of money and consumers trust. Dependability is then an important aspect of the cloud and crucial for building reliable cloud services. Providers are focused on availability and performance so the existing benchmarks mainly evaluate those characteristics. Dependability is hard to benchmark in cloud computing because of its complexity and dynamic changes overtime. The lack of redundancy caused by the monopolization of the cloud, affected the cloud's self-healing ability and providers began to experience outages and this led to the interest in measuring dependability.

In this paper, we study the following cloud benchmarks: Spec IaaS 2016, TPCx-V, YCSB, Perfkit Benchmarker, CloudBench, DS-Bench/D-Cloud, and evaluate if they are suitable for benchmarking dependability.

The rest of the paper is organized as follows. In Section 2 we describe dependability and faults. In Section 3 we present existing benchmarking tools describing them and assessing dependability characteristics. In Section 4 we summarize the dependability attributes benchmarking. Finally, Section 5 concludes the paper and propose future work.

Carvalho, E., Barbosa, R. and Bernardino, J.
The Ability of Cloud Computing Performance Benchmarks to Measure Dependability.
DOI: 10.5220/0006472404470452
In *Proceedings of the 12th International Conference on Software Technologies (ICSOFT 2017)*, pages 447-452
ISBN: 978-989-758-262-2

2 DEPENDABILITY

Dependability can be defined as the ability to guarantee that a service will accomplish its functions. Dependability emphasizes trust and reliability, for a cloud to be dependable it must guarantee its services are always available and allow consumers to operate without having problems. To benchmark dependability, we must ensure its attributes are respected (Avizienis et al., 2004):

- Availability: readiness for correct service;
- Reliability: continuity of correct service;
- Safety: absence of catastrophic consequences on the user(s) and the environment;
- Integrity: absence of improper system alterations;
- Maintainability: ability to undergo modifications and repairs.

The main goal of dependability is to mitigate the chain of faults that may lead to errors that can propagate and lead to failures which can cause more faults. To do that the system must have (Gainaru and Cappello, 2015):

- Fault prevention: be able to prevent faults from happening, be robust;
- Fault tolerance: be able to avoid or mitigate service failures in the presence of faults;
- Fault removal: be able to recover from faults reducing then the number of faults present in the system;
- Fault forecasting: be monitored to detect existing faults, be able to map when similar faults happen and estimate the consequence of those faults.

Dependability is hard to measure and compare because each system has different failure modes and faults are difficult to inject due to security of privileged layers of the cloud. However, there are ways to inject faults in the virtualization and hardware layer. Fault tolerance and fault removal can then be evaluated by benchmarks with the help of proper metrics such as impact on performance and time to recover. To achieve fault prevention and forecasting, the system should be monitoring the existing faults for analysts to understand the root of the faults and how they propagate between layers. Failure monitoring becomes, then, an important part of the process of evaluating dependability, if a failure happens it means that faults couldn't be contained and the above standards are not being respected. A failure detection metric is suggested in (Guan et al., 2012), it consists in measuring *Precision* (Number of correctly detected failures per Total number of failures); *Sensitivity* (Number of correctly detected failures per Total number of detected failures); *Specification* (Number of correctly detected normal records per Total number of normal records) and *Accuracy* (Total number of correct detections per Total number of data records). In the following section, we present benchmarking tools for performance evaluation that together with new metrics such as the ones presented above, could be used to measure dependability.

3 BENCHMARKING TOOLS

We selected some of the most used benchmarks for cloud computing to evaluate if they cover dependability. In this section, we present how each benchmark works, the metrics they use and more important, the answer to the question "can they be used to benchmark dependability?".

3.1 Spec IaaS 2016

Spec IaaS 2016 benchmark was designed by SPEC to measure the performance of public, private or hybrid IaaS clouds using a representative real-workload (https://www.spec.org/cloud_iaas2016/). For this purpose, Spec uses YCSB with Cassandra and K-Means implementation from HiBench as the benchmark workloads. These workloads are managed by a benchmark harness, Cloud Bench (cbtool), this harness is responsible for creating and destroying instances, initiate application instances (which are a set of instances) and collect measurements and data points as well as computing the scores for each submission. SPEC uses YCSB for I/O intensive testing, in more detail YCSB workload D (95% read, 5% insert) to simulate social network user's activities. Apache Cassandra is the NoSQL database used. For compute-intensive workload SPEC uses one of the nine Hadoop workloads, K-Means. SPEC reports eight metrics: Elasticity, Scalability, Mean Instance Provisioning Time, AI Provisioning Success, AI Run Success, Total Instances, Elasticity Start Time and Elasticity End Time. Elasticity is the average of the elasticity measured by YCSB and K-Means, the performance of N applications in the elasticity+scalability phase must be similar to the elasticity in the baseline phase where the load is constant. Scalability measures the contribute of N applications instances compared to the contribute of one application instance for a successful cloud it should scale linearly. Mean instance provisioning time measures the time it takes

the system under test to provision valid application instances. The AI Provisioning Success measures the amount of successfully provisioned AIs in percentage. AI Run Success measures the percentage of successful runs. Elasticity Start and End time indicate when the elasticity starts and ends respectively. Total instances indicate the number of instances provisioned during the benchmark. This benchmark covers mostly performance and availability presenting no means to evaluate safety, integrity and maintainability.

3.2 TPCx-V

TPC Express Benchmark™ V (TPC Express Benchmark™ V - Specification, Revision 1.0.1, 2016) is one of TPC's benchmarks for databases running in a virtualized environment. Its benchmarking kit is public-available in (http://www.tpc.org/tpcx-v/default.asp). This benchmark measures the performance of the hypervisor, hardware, storage and networking of a server running virtualized databases, but it differentiates itself because it is able to model many cloud proprieties, such as running VMs with different load demands and varied load fluctuations thus testing scalability and elasticity. The workload consists of On Line Transaction Processing and Decision Support Systems and the transaction mix can be found in (TPC Express Benchmark™ V - Specification, Revision 1.0.1, 2016). The primary metrics of TPCx-V are the Reported Throughput expressed in number of valid transactions per second, it then takes the total 3-year pricing and divides it by the reported throughput for a price/performance metric. Like TPC-W it enforces ACID transactions and has integrity rules, such as Referential Integrity (TPC Express Benchmark™ V - Specification, Revision 1.0.12016). To measure elasticity, the CPU usage is measured while running the transactions and compared with a run where elasticity isn't configured. It also measures maintainability by running a Data-Maintenance Transaction and checking how many transactions are done in a certain response time.

3.3 YCSB – Yahoo! Cloud Serving Benchmark

Yahoo! Cloud Serving Benchmark (Cooper et al., 2010) is an open source framework designed to evaluate NoSQL databases. This benchmark provides a data generator and a set of tests that perform specific of operations over the databases, called workloads (Abramova et al, 2014). The standard operations are: create, read, update and delete. To execute the YCSB benchmark there is a tool called YCSB Client which offers extensibility so that we can create different workloads. This benchmark can be used to evaluate performance and scalability of cloud systems' NoSQL databases, it can do so by injecting read/write heavy workloads. The YSCB has six pre-defined workloads (https://github.com/brianfrankcooper/YCSB):

- Workload A: 50% reads, 50% updates.
- Workload B: 95% reads, 5% updates.
- Workload C: 100% reads.
- Workload D: 95% reads, 5% inserts.
- Workload E: 95% scans, 5% inserts.
- Workload F: Read-Modify-Write.

These loads are driven by four different distributions which chose what record to read or write and what operation to perform: Uniform, Zipfian, Latest, Multinomial. The metrics used by YCSB are: latency vs throughput for performance; for scaling it uses a scale up and elastic speedup metric, as the load increases the performance should remain the same, if the system has a good scale up, for the elastic speedup as the same workload is running one or more servers are added and the performance should increase; For availability YCSB measures the impact on performance as a server is killed while the workload is running; Replication, this tier of evaluation measures the impact of replication on performance and availability. Replication can be synchronous or asynchronous, the first prevents data loss when replicating for example in case of a failover, the second may cause loss of data if a failure happens before the replication is scheduled but improves performance.

3.4 Perfkit Benchmarker

PerfKit Benchmarker is an open-source benchmarking tool used to measure and compare cloud offerings According to its website (http://googlecloudplatform.github.io/PerfKitBenchmarker/) it is a community effort involving over 500 participants including researchers, academic institutions and companies together with the originator, Google. Perfkit values for its simplicity as it can run tests to any SSH able machine just by installing the benchmark, its requisites and running a set of commands or config file. The code to Perfikit is on Github.com which is a way of showing the public that the code can be trusted and doesn't favour any cloud provider, for example Google who

developed the benchmark. The benchmark can initiate virtual machines with the selected benchmarks in the selected cloud provider or run the benchmarks in a local machine. Perfkit uses YCSB's Aerospike, Cassandra, Hadoop Terasort, HBase, MongoDB and Redis, these are NoSQL databases very often used in the cloud. For high performance computing, it uses HPCC (High Performance Computing Cluster). As for simulation workload, it uses OLDIsim to measure the scaling capability of the system. Then it has some minor benchmarks that evaluate the hardware: for CPU it has Coremark and Spec cpu 2006, for disk it has Bonnie++, Copy, Fio, Synthetic Storage and for network it has Iperf, Mesh, Network, Netperf and Ping. It has Unixbench and Clusterboot as system benchmarks to evaluate the performance of software and hardware working together. The main metric of Perfkit Benchmark is end-to-end time to provision which consist of measuring the time of the following phases (Filho, 2015): Setup, Warm up, Pre-execute, Execute, Post-Execute, Clean-up and Publish results. Setup and Warm up is the time that the cloud provider will take to provide the VM's and install the benchmarks. Pre-execute is the time it takes to load the data for the benchmarks, execute is the time it takes to run the benchmarks. Post-Execute is data analyses; Clean-up is the time it takes to stop all the services deployed by the cloud, because leaving the benchmark accidentally running could get very expensive and finally publishing the results that can be viewed in Perfkit Explorer. The end-to-end time to provision metric is complemented by the metrics reported by the benchmarks Perfkit uses. Cloud Harmony is also a framework like Perfkit which allows the users to launch various test on cloud provider, however it doesn't have a metric of its own and it is not open-source.

3.5 Cloud Bench

Cloud Bench (Silva et al., 2013) is an open-source framework that runs benchmarks in the cloud through the deployment of complex applications. It can run experiments in multiple clouds across various regions using a single interface. The application deployment is automated and occurs in this order: VM creation, application configuration, controlled execution, data collection and VM termination. This benchmark can only be used if the cloud is capable of doing this task without human interaction. CloudBench uses Virtual Applications or VApps as workloads which can be managed by it and can run the benchmark in multiple VMs. A VApp is defined by type, topology, configuration steps and load behaviour. The type is the benchmark to be run, the topology is the amount of VMs needed for it, the configuration steps are the scripts run in in each VM to setup the benchmark and the load behaviour is the variation of the load to be applied to the VApp. While the VApps run, a number of data is collected and reported as metrics such as VM provisioning time, failure and time to recover, time to scale and adapt to load increased/decreases and standard runtime metrics such as measuring throughput, latency and bandwidth.

3.6 DS-Bench/D-Cloud

DS-Bench/D-Cloud (Ishikawa et al., 2012) is a benchmark designed to execute dependability evaluations in virtualized and physical environments by measuring availability, reliability performance and power consumption under an anomaly situation. These anomaly situations can be hardware faults, software faults or human errors. D-Cloud is a tool to test the system by managing resources. This benchmark is easy to use and comes with a GUI where we can create a benchmark scenario and build a script for the anomaly scenario as well. This benchmark works with two controllers DS-Bench controller and D-Cloud controller, the DS-Bench controller receives the benchmark configurations and communicates them to D-Cloud who then starts the benchmarking process by requesting the cloud to setup the VMs with the selected benchmarks (see Figure 1).

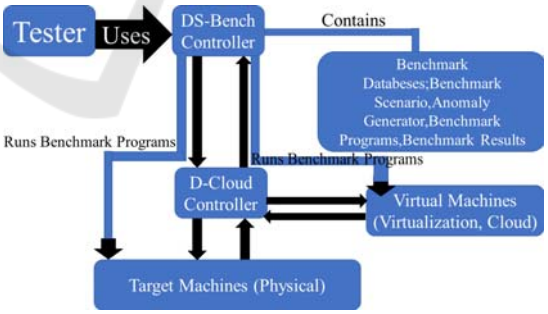

Figure 1: DS-Bench/D-Cloud setup example. Adapted from (Ishikawa et al., 2012).

4 DEPENDABILITY ANALYSES COMPARISON

In this section, we discuss the results of our research by summarizing the dependability characteristics of

the benchmarks presented before. Cloud computing is a very complex set of hardware and software and in this type of system failures are common.

It is of extreme importance to prevent and recover from failures quickly, as there are many services depending on the proper functioning of the cloud, and to do this we need to study and understand how they occur in the cloud environment. A failure in the cloud can mean a significant profit loss not only to the provider but more importantly to the users. A failure can take catastrophic proportions, such as denial of service to clients, which can bring harm to the user. For example, if the user is in a self-driven car that uses cloud services for guidance, and that is why when benchmarking the cloud, the benchmarks should measure not only fault tolerance but also fault prediction and recovery. To predict faults, we can have a set of metrics that calculate based on the pattern of software and hardware faults, when the next fault will possibly happen.

Performance benchmarks like the ones we present, fit in the category of *microbenchmarks* referenced in (Oppenheimer et al., 2002) because they assess some of the dependability characteristics individually and not the end-to-end dependability of the system. To evaluate dependability as a *macrobenchmark*, the benchmark should possess an integrated fault load and fault load injection tool to be deployed as we run performance benchmarks as workloads. The fault load should be realistic and representative of real world hardware, software and human made faults. Cloud is a large system, where many fault scenarios can happen which makes it difficult for a benchmark to analyse them all and also be able to repeat them in other cloud systems. According to Oppenheimer et al. (2002) the studies on Internet services, large servers, and the public telephone network indicate that human error is the largest single cause of service unavailability. However, having a fault simulator of human made faults is essential but given the differences from cloud to cloud it is very difficult to create a generic script that covers all human made faults and replicate them in different cloud systems. Another concern is keeping the benchmark valid as some cloud systems may detect they are under test and try to trick the results by preventing the benchmark induced faults. For this reason fault selection must be studied and changed overtime which invalidates previous benchmark tests and those have to be done again. Running a dependability dedicated benchmark in a production environment with frequency becomes expensive because the benchmark takes a long time and most of the time requires human interaction to select and inject faults. Therefore one of the solutions is benchmarking attributes of dependability to reduce costs.

Most of the presented benchmarks lack a fault load and evaluate only some aspects of dependability. Availability and reliability are given characteristics measured by the benchmarks because most of them have a price vs performance metric.

Table 1: Comparison between benchmarks.

	Availability	*Reliability*	*Integrity*	*Maintainability*	*Safety*
Spec Iaas 2016	Yes	Yes	No	No	No
TPCx-V	Yes	Yes	Yes	Yes	No
YCSB	Yes	Yes	No	No	No
Perfkit Benchmarker	Yes	Yes	No	No	No
Cloud Bench	Yes	Yes	No	Yes	No
DS-Bench/D-Cloud	Yes	Yes	Yes	Yes	No

Table 1 presents these characteristics for all the presented benchmarks. Under normal conditions, *Availability* and *Reliability* exists in the systems because the performance is not null. However, it should be tested under a fault load to measure the impact of failures in the system.

Integrity however is only measured by DS-Bench/D-Cloud by checking the hard-disk for errors and if the received data contains errors. This is a very important characteristic to evaluate as some applications may not be able to function if data is corrupt, leading to system alterations and malfunction. *Maintainability* is only evaluated by some benchmarks and it's also an important characteristic as it guarantees the system functionality. To measure maintainability DS-Bench/D-Cloud and CloudBench run benchmarks while a fault load is injected. However TPCx-V uses a data-maintenance transaction and measures how many transactions are done in a given time interval. Metrics like time to recover and throughput vs latency should be applied by benchmarks while the system is under the influence of a fault load.

Safety is not evaluated by any of the benchmarks and despite not being of interest now it can become one in a near future. As self-driven cars are already getting their system running in the cloud and a failure could indeed cause harm to the user or the environment. As future work, we want to use the benchmarks to generate workload and complement them with new metrics to measure dependability while running a fault load during the benchmark.

5 CONCLUSIONS AND FUTURE WORK

With companies that depend on cloud computing to sustain their business it is critical to have a benchmark that can help the decision makers evaluate if the cloud provider they are choosing can meet their requirements.

In this paper, we have presented a variety of benchmarks and the metrics they use when evaluating the cloud. We can conclude from our study that the existing benchmarks for cloud computing focus on measuring performance and availability comparing the results with the monetary cost and do not have strong metrics to measure all of the dependability characteristics.

Most of the benchmarks lack fault simulation or injection because they do not possess a fault load or metrics to do so, which would allow to measure maintainability. Integrity is also not measured by benchmarks as they focus on measuring I/O disk operations and database interactions per second and lack a metric to measure data integrity.

Although TPCx-V provides integrity by enforcing ACID transactions and integrity rules. Safety is not contemplated at all by the benchmarks as it is not yet an important characteristic to be considered in cloud environments as the cloud's malfunction does not have a serious impact on the environment or user.

As future work, we propose to integrate a fault load and new metrics so that dependability becomes a part of the new generation of benchmarks for cloud systems to help providers and consumers evaluate the impact of possible failures.

REFERENCES

Abramova, V., Bernardino J., Furtado P., 2014.Testing Cloud Benchmark Scalability with Cassandra. In *IEEE World Congress on Services*, Anchorage, AK, 2014, pp. 434-441.

Abramova V., Bernardino J., Furtado P., 2014. Evaluating Cassandra Scalability with YCSB. In Decker H., Lhotská L., Link S., Spies M., Wagner R.R. (eds) *Database and Expert Systems Applications. DEXA 2014.* Lecture Notes in Computer Science, vol 8645. Springer, Cham.

Avizienis, A., Laprie, J., Randell, B., Landwehr, C., 2004. Basic Concepts and Taxonomy of Dependable and Secure Computing. *In IEEE Transactions on Dependable and Secure Computing.*

Cooper, B., Silberstein, A., Tam, E., Ramakrishnan, R., Sears, R., 2010. Benchmarking Cloud Serving Systems with YCSB. In SoCC *'10 Proceedings of the 1st ACM symposium on Cloud computing*, pp. 143-154.

Filho, I., 2015. PerfKit Benchmarking the Cloud. *In CouchBase Connect.*

Gainaru A., Cappello F., 2015. "Fault and failures". *Chapter 2 of Fault-Tolerance Techniques for High-Performance Computing, Springer Book, Computer Communications and Networks series*, Editors: Thomas Herault and Yves Robert.

Guan, Q., Chiu, C., Fu, S., 2012. CDA: A Cloud Dependability Analysis Framework for Characterizing System Dependability in Cloud Computing Infrastructures. *In IEEE 18th Pacific Rim International Symposium on Dependable Computing*.

Ishikawa, Y., Sato, M., Hanwa, T., Fujita, H., Banzai, T., Koizumi, H., Miura, S., 2012. DS-Bench/D-Cloud: Dependability Measurement and Evaluation Tool. *In 42nd IEEE/IFIP International Conference on Dependable Systems and Networks (DSN 2012).*

Khan M.,6 April 2017, Yahoo! *Cloud Serving Benchmark(YCSB)*, Available from: https://github.com/brianfrankcooper/YCSB, (20 May 2017).

Neves, P., Bernardino, J., 2015. Big Data in the Cloud: A Survey. *In Open Journal of Big Data (OJBD)*, 1(2), Pages 1-18, RonPub. [Permalink]: http://www.ronpub.com/publications/ojbd/OJBD_2015v1i2n02_Neves.html.

Neves, P., Schmerl, B., Cámara, J., Bernardino, J., 2016. Big Data in Cloud Computing: Features and Issues. *In Proceedings of the International Conference on Internet of Things and Big Data – Volume 1: IoTBD*, ISBN 978-989-758-183-0, pages 307-314.

Oppenheimer, D., Brown, A., Traupman, J., Broadwell, P., Patterson, D., 2002. Practical issues in dependability benchmarking. *In Workshop on Evaluating and Architecting System dependability (EASY '02).*

Perfkit Benchmarker Authors, 2016, *PerfkitBenchmarker*, Available from: http://googlecloudplatform.github.io/PerfKitBenchmarker/, (25 May 2017).

Standard Performance Evaluation Corporation, 3 January 2017, *SPEC Cloud™ IaaS 2016*, Available from: https://www.spec.org/cloud_iaas2016/, (17 May 2017).

Silva, M., Hines, M., Gallo, D., Liu, Q., Ryu, K., Silva, D., 2013. CloudBench: Experiment Automation for Cloud Environments. *In IEEE International Conference on Cloud Engineering.*

Transaction Processing Performance Council, 2017, *TPCx-V*, Available from: http://www.tpc.org/tpcx-v/default.asp, (15 May 2017).

Transaction Processing Performance Council, 2016, *TPC Express Benchmark™* V - Specification, Revision 1.0.1.

Vazquez, C., Krishnan, R., John, E., 2014. Cloud Computing Benchmarking: A Survey. *In International Conference on Grid & Cloud Computing and applications.*