Exam Cloud Computing

Sample Exam, December 2019

Examiner: dr. K. F. D. Rietveld

- This exam is **closed book**, so it is not allowed to use the textbook, (printed) slides or your own class notes.
- Only calculators that do not have the possibility to store text files may be used. Use of graphing calculators is **not** allowed.
- The questions must be answered in English.

- When handing in your work you will be requested to note your name, student ID and the number of pages that you are handing in on the attendance list.
- *The question sheet must be handed in at the end of the exam.*

- The number of problems is 4 with a total of 15 items. The exam consists of 4 pages.
- Behind each item the number of points that can be scored is shown between square brackets. The total amount of points that can be scored is 100.
- The obtained grade counts towards the final grade for the course with a weight of 35%.

Problem I: Fundamentals [25 points]

a. [8 points] Briefly describe the three service models of cloud computing (SaaS, PaaS, IaaS) **and** give a concrete example service for each of these.

b. [6 points] For security-sensitive applications, one usually prefers a hypervisor-based deployment over a container-based deployment. Why is this the case?

c. [6 points] Explain how a service level agreement (SLA) can be used for optimization from the perspective of both the customer as well as the provider.

d. [5 points] What is the general concept of NFV (Network Function Virtualization)?

Problem II: Understanding Literature [25 points]

Read the following excerpt from the Introduction of the paper:
Carsten Binnig, Donald Kossmann, Tim Kraska, and Simon Loesing. 2009. How is the weather tomorrow?: towards a benchmark for the cloud. In Proceedings of the Second International Workshop on Testing Database Systems (DBTest '09). ACM, New York, NY, USA, Article 9, 6 pages. `http://dx.doi.org/10.1145/1594156.1594168` [1]

Traditionally, the goal of benchmarking a software system (called system under test or short SUT) is to evaluate its average performance under a particular workload. The most prominent examples for evaluating transactional database systems as well as other components on top (such as a application-servers or web-servers) are the various TPC-benchmarks that define workloads derived from different real-world application scenarios (e.g., TPC-H for OLAP, TPC-C for OLTP, or TPC-W for an e-commerce application).
All the TPC-benchmarks require that the system under test is deployed in a managed environment using a fixed configuration (e.g., a static set of software- and hardware-components) which is described in a full disclosure report. Consequently, the primary metrics of all the TPC-benchmarks reflect the average performance of a static non changing system. Another primary metric that is often reported by the TPC-benchmarks are the total costs of ownership for such a static system over its lifetime (i.e., the costs for hard-and software, maintenance as well as administration). Moreover, as all TPC-benchmarks focus on transactional database systems they force these systems to provide the ACID properties as well as different levels of isolation as defined by the SQL standard.
[...]
Another important difference of the cloud storage services compared to the traditional transactional database systems are the consistency guarantees provided by these services. In order to offer fault-tolerance and high-availability, most providers replicate the data within one data or even across data centers. Following the CAP theorem, it is not possible to provide availability and strong consistency (as defined by the ACID properties) together in the presence of network failures. Consequently, most cloud providers sacrifice strong consistency for availability and offer only some weaker forms of consistency (e.g., Amazonss S3 guarantees only eventual consistency).

[1] Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. DBTest09, June 29, 2009, Providence, Rhode Island, USA.Copyright 2009 ACM 978-1-60558-706-6/09/06 ...$5.00.

a. [5 points] The paper mentions the "eventual consistency" model. Explain in your own words what this means and how it differs from strong consistency.

b. [10 points] Discuss at least two properties of cloud systems that are not considered by traditional benchmarks such as TPC-H, TPC-C and TPC-W, as mentioned in the text, and benchmarks like SPEC. Explain why these traditional benchmarks do not consider these properties and describe how you think these properties should be benchmarked?

c. [4 points] The paper mentions in its abstract that 'different providers' provide 'different guarantees'. One such guarantee concerns consistency (which is in fact an implementation detail of a particular service). Considering a cloud provider as a whole, what other differences in guarantees can you think of?

Later on in the paper the authors write: *As imposing one architecture variant [of cloud services] is not reasonable, a cloud benchmark should be general enough to cover the different architectural variants. Furthermore, the complete application stack should be measured instead of micro-benchmarking single services.*

d. [6 points] Do you think a case can still be made for the micro-benchmarking of single services? If yes, explain why and in what scenario would such a benchmark still be useful? If no, explain why not.

Problem III: Design Scenario [25 points]

Santa Claus runs software for the management of wish lists on a private cloud. Wish lists are stored in a database and can be edited through a web-based front-end. The system also provides an analysis backend that is used to find opportunities for bulk purchases to be able to negotiate discounts from suppliers.
Analysis of the traffic to this application has shown that the traffic pattern is very seasonal. Therefore, Santa Claus is looking into a hybrid cloud setup with cloud bursting to be able to withstand traffic peaks.

a. [4 points] Describe what is understood by Cloud Bursting.

An essential component for a bursting system is a cloud broker, which can communicate with multiple different architecture providers and which is to decide what provider to lease resources from.

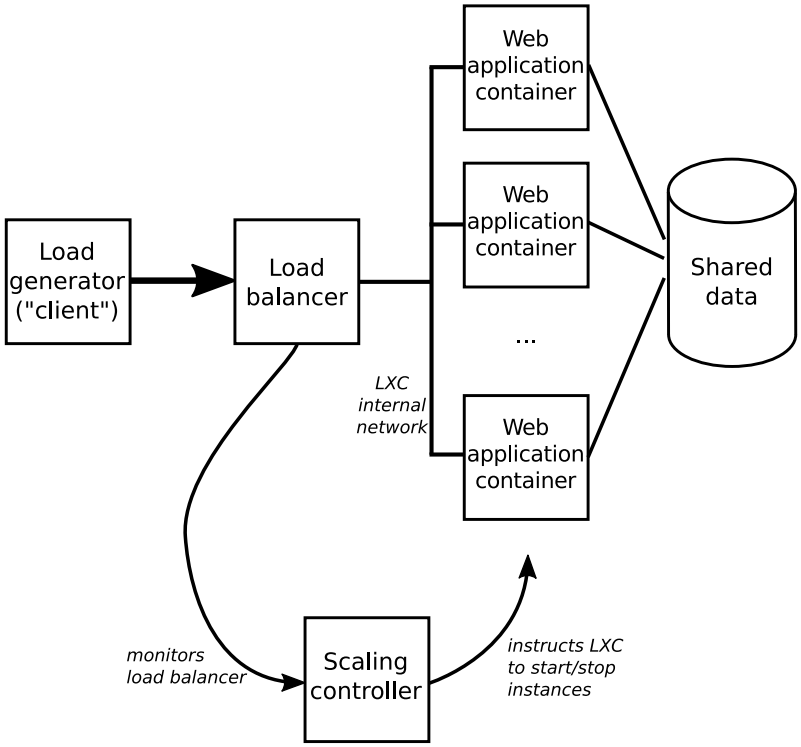
b. [5 points] With the current state of affairs, what is the main difficulty in the implementation of a broker that supports different providers?

c. [4 points] We described and discussed different pricing models for IaaS during class. Given Santa's very seasonal (and recurring) traffic pattern, what pricing model could you take advantage of in order to reduce cost?

d. [12 points] If you were consulted to determine whether Santa's current application and application architecture would be suitable for cloud bursting, what items would be on your check list and why? (We expect at least four concrete items).

Problem IV: Lab Assignment [25 points]

The first lab assignment concerned the development of a small autoscaling object store application based on Linux containers (LXC). The general overview of the developed system is shown in the following figure:



a. [6 points] For the container images it was strongly suggested to use "copy-on-write". Clearly explain:

1. What is meant by 'copy-on-write'?
2. Why this was suggested. What would be the consequence if this property was not used?

Load balancing was done by means of the round robin methodology. One could also imagine to balance the load by partitioning the data. So, this implies that each object store instance serves a different partition of the data.

b. [7 points] What is a potential problem with this approach with respect to the effectiveness of load distribution and how can this be solved?

c. [12 points] Describe what changes are required to the architecture shown in the figure to implement this approach. Consider all components of the system and also discuss the changes necessary to make up and down scaling work correctly. It may be useful to include a diagram in your answer.