Future Generation Computer Systems 78 (2018) 115–126

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

CloudStore — towards scalability, elasticity, and efficiency benchmarking and analysis in Cloud computing

Sebastian Lehrig [a,*], Richard Sanders [b], Gunnar Brataas [b], Mariano Cecowski [c], Simon Ivanšek [c], Jure Polutnik [c]

[a] s-lab – Software Quality Lab, Zukunftsmeile 1, 33098 Paderborn, Germany
[b] SINTEF, Strindvegen 4, 7043 Trondheim, Norway
[c] XLAB, Pot za Brdom 100, 1000 Ljubljana, Slovenia

HIGHLIGHTS

- CloudStore is introduced as a novel open-source benchmark for cloud computing applications.
- Novel cloud computing metrics are evaluated for CloudStore operating in Amazon EC2.
- Support for the novel metrics is integrated in a performance model of CloudStore.

ARTICLE INFO

Article history:
Received 31 March 2016
Received in revised form
18 December 2016
Accepted 11 April 2017
Available online 25 April 2017

Keywords:
Cloud computing
Measurements
Scalability
Performance
Capacity
Elasticity
Efficiency
Amazon Web Services
TPC-W

ABSTRACT

This paper describes CloudStore, an open source application that lends itself to analyzing key characteristics of Cloud computing platforms. Based on an earlier standard from transaction processing, it represents a simplified version of a typical e-commerce application – an electronic book store. We detail how a deployment on a popular public cloud offering can be instrumented to gain insight into system characteristics such as capacity, scalability, elasticity and efficiency. Based on our insights, we create a CloudStore performance model, allowing to accurately predict such properties already at design time.

1. Introduction

Cloud Computing has and is gaining traction in the ICT industry since the turn of the millennium [1]. Hybrid, private and public clouds promise a number of benefits for enterprises. However, there does not seem to be a range of standardized applications which can be used to compare the performance of different cloud deployments. In particular, there is a lack of standard open sourced applications to express and compare metrics.

To this end the CloudScale project [2] has developed an application we call CloudStore [3]. CloudStore is a free and open implementation of a typical web application, with some inherent scalability and performance issues related to the relational database. The goal of CloudStore is to serve as a relevant and standardized baseline to measure and compare different cloud providers, architectures and deployment in terms of capacity, scalability, elasticity and efficiency. CloudStore embodies functionality for an online book store, and is based on the transactional web e-Commerce benchmark (TPC-W), a web server and database performance benchmark originally proposed by the Transaction Processing Performance Council [4]. Although now obsolete, TPC-W has functional and non-functional characteristics that can be used to achieve our aims.

CloudStore and the measurements methodology help answer questions such as "which architecture makes more efficient use of

* Corresponding author.
E-mail addresses: Sebastian.Lehrig@uni-paderborn.de (S. Lehrig), Richard.Sanders@sintef.no (R. Sanders), Gunnar.Brataas@sintef.no (G. Brataas), Mariano.Cecowski@xlab.si (M. Cecowski), Simon.Ivansek@xlab.si (S. Ivanšek), Jure.Polutnik@xlab.si (J. Polutnik).

http://dx.doi.org/10.1016/j.future.2017.04.018
0167-739X/

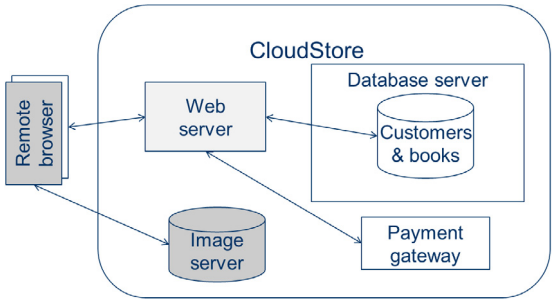

Fig. 1. Conceptual CloudStore architecture.

our resources?", "which cloud provider is cheaper for an expected usage fluctuation?", "which implementation/deployment is more scalable within our expected range?"

Although our motivation for creating CloudStore was the need for an open application to demonstrate the scalability tools of the CloudScale project [2], CloudStore can equally well be used for comparing (benchmarking) different cloud and platform providers (Amazon, Google, Microsoft, and private clouds). Benchmarking is generally costly. A standardized service like CloudStore may reduce such costs.

The present paper extends our previous publication [5] by a performance model for CloudStore and a related work discussion; the first part of this paper is almost identical to [5] except for some minor improvements. The performance model promises to reduce analysis costs compared to benchmarking. With such models, software architects can assess different deployment scenarios via performance analyses. These can in general be executed with less effort than conducting benchmarks.

This paper is structured as follows. Section 2 gives an overview about CloudStore. Afterwards, we describe how we deployed CloudStore within the Amazon Web Services in Section 3. Based on this deployment, we use CloudStore in Section 5 to benchmark various metrics that assess CloudStore's capacity, scalability, elasticity, and efficiency. Section 6 leverages analyses for such metrics on the model level by deriving a performance model for CloudStore. We discuss related work in Section 7 and conclude this paper with a summary and a discussion of potential future work in Section 8.

2. CloudStore—a generic e-commerce application for Cloud computing

CloudStore is a typical example of a web application. Its structure is depicted in Fig. 1. A user makes use of the application via a web browser, which has direct IP communication to components that provide the user experience. Most of the functionality is handled by a Web server, which receives HTTP requests from Remote browsers, and for information processing purposes connects to a Database server storing information about users (customers) and the items of the store (information about books). The HTTP response back to the Remote browser contains references to an Image server (to efficiently display pictures of books and other graphics). In the case of payment transactions the HTTP response contains references to a Payment gateway. TPC-W specifies that the system should obtain credit card authorization from a Payment Gateway Emulator, and, if successful, present the user with an order confirmation.

The original TPC-W standard [4] on which CloudStore is based consists of a detailed specification of an on-line shop, defining 14 operations for browsing and buying books, together with non-functional requirements such as data element size and time responses, as well as expected user behavior and work load. The standard was originally designed for benchmarking implementations and deployments of transaction systems. We have taken the specifications to create an application that provides us the means to measure characteristics such as capacity, scalability, elasticity, and efficiency in alternative deployments using cloud computing.

In addition to re-implementing the functional behavior, we adopted the non-functional requirements, such as the percentage of errors and time responses, in order to produce Service Level Objectives (SLOs) that should be respected by the deployment to be tested. The result is a scenario for evaluating, e.g., scalability, elasticity, and actual costs of different cloud solutions using a simple yet realistic example of an e-commerce application.

TPC-W defines three different usage profiles: a shopping mix, browsing mix and ordering mix. For each usage profile a different probability rate is defined for each of the 14 operations.

3. CloudStore methodology

In order to use CloudStore to generate measurements for the analysis of a platform, infrastructure, or deployment, a plan and some definitions are necessary. Namely, we need to define the aspects that are to be kept constant, and which will variate. Generally speaking, if we are comparing for example two PaaS environments, we will keep the implementation of CloudStore, the data in the system, the load generation, and other aspects unrelated to the PaaS environment completely unchanged. CloudScale provides tools to create data for the application, as well as a distributed Jmeter implementation for generating the load. Finally, we need to define the metrics that we want to obtain and what data we need to log during the tests to obtain these metrics.

For each of the test-runs to be performed, and keeping all other variables constant, we need to:

- deploy and instantiate the application in the given environment,
- generate the load that will stress-test the application,
- gather the resulting information from the test-run, and
- calculate metrics.

4. CloudStore deployed on Amazon Web Services

To exemplify how CloudStore can be used, we show how we have deployed and instrumented it using a public cloud provider, in this case the widely popular Amazon Web Services (AWS) [6]. To that end, we will be using some PaaS solutions provided by AWS such as the RDS database and the S3 static content, but a deployment could be also performed on the OpenStack IaaS with manually deployed database and static storage services, or even deployed to an Application Server service like SAP's HANA Cloud PaaS environment. Embarking on this, we made some design choices, both regarding how the functional components were deployed, how to generate synthetic user traffic and how to measure key characteristics such as resource utilization, response times (including violations) and cost.

4.1. The deployment architecture

Fig. 2 shows one deployment of CloudStore with the load generator (Distributed JMeter), the functional components of CloudStore (Web server, Database server, Image server and Payment gateway) and their deployment on the Amazon cloud computing platform. The following is a brief account of the deployment and what it entails.