Cyberspace: Jurnal Pendidikan Teknologi Informasi,
Volume 1, Nomor 2, Oktober 2017, 94-100

SCHEDULED BASED CLOUD RESOURCE ALLOCATION

Hendy Mizuardy

Pendidikan Teknologi Informasi, Fakultas Tarbiyah dan Keguruan
UIN Ar-Raniry Banda Aceh – Indonesia, 23111
Email: hendy.mizuardy@ar-raniry.ac.id

Abstract

The objective of this research the tremendous implementation of cloud computing technology has become a new trend that users can easily utilize high resources through IaaS platform. IaaS is more economical and easier way to have physical resources; in this case Virtual Machines in the cloud, rather than building the infrastructure by their own. To deliver internet services to users such as website, email service or other software applications, a service provider can utilize IaaS platform by leasing virtual infrastructure from a cloud provider and deploy their services on that VMs. However, it becomes a challenge for a service provider to maintain their services due to the increasing number of user requests. They have to maintain resources availability to provide maximum performance to meet their user satisfaction with optimal resources utilization. The approach in this paper will solve this problem by providing service provider a resource monitor module. The module monitors VMs workload based on schedule approach; peak time and off-peak time. According to these two criteria, the service provider can predict and allocate sufficient resources.

Keywords*:* *Data Center, Resource Monitoring, Virtual Machine.*

1. Introduction

The advancement of technology in broadband networks, web services, computing systems, and applications has created a massive change in cloud computing concept. As a result, cloud computing has become a new technology trend in which users can easily access high computer resources [5]. To obtain resources needed by users, cloud service providers can either set up their own infrastructure or use Infrastructure as a Service (IaaS) platform. Of those two methods, using IaaS is more economical and easier than build their own infrastructure. They can use API provided by IaaS providers, such as Amazon Web Service (AWS) or Google Compute Engine (GCE), to create and configure Virtual Machine, disk, memory, load balancer, etc. By paying the resources as they used, it will save the cost a lot. However, it becomes a challenge for service providers to maximize the performance and minimize the financial cost (Lee, *et.al*, 2010). Auto scaling mechanism is one approach used in IaaS in which service providers can maintain the resources and reduce waste resources by automatically increase or decrease them when needed.

Some of the cloud services support auto-scaling functionality. They can monitor information about CPU utilization, disk I/O and network I/O on a server side to trigger system configuration [5]. Yet, it is still difficult to predict for client side which later affects in decreasing performance because of the lack of computing instances. To solve this problem, schedule based resource allocation proposes a method to monitor system workload. In this case, the service providers can predict peak time or off-peak time and then prepare such a sufficient resources. The proposed architecture can be adopted by service providers to evaluate their system performance before releasing their services. The rest of the paper is organized as follows. We will discuss related work in section related work. Section proposed architecture describes the proposed model, while section evaluation and discussion, evaluates and discusses the solutions. Finally, section conclusion and future work, concludes the proposal and discusses some future works.

2. Related Work

To minimize the cost and to satisfy the requirement of performance are two most important issues for cloud service providers. Workload monitoring plays the main role in providing enough resources to satisfy the demanded capacity while reducing the waste of resources itself. There are some approaches have been conducted to measure the cloud server performance. In [1], it uses measured capacity of VM instance and arrival request rate to estimate the response time or cumulative distributions of the response time on a certain number of VM instance. But these approaches cannot generally adapt to different service architecture. Because the system may be composed of many different other services, to get all resources detail capacity may be impossible.

There is an SLA-driven system [2] that just needs to set a request processing time between load balancer and application servers. And the load balancer in front of server nodes checks the server-side response time. In summary, current related works have a problem in determining a good indicator to do resource scaling. To solve these problems, in this work, we proposed an approach to monitors the cloud server workload in peak-time and peak-off time to provide information for a service provider to do schedule-based scaling to give a better performance for the user in the peak-time and reduce the resource waste in the peak-off time. This approach can also be applied to many architectures because it is based on the peak and peak-off time. The comparison among those approaches is shown in Table 1 below.

Table 1. Related Work

	Proposed (This Paper)	**Related Work [1]**	**Related Work [2]**
Testing Tool	Locust	AMS CloudWatch	Nginx
Cloud Server	GCE	Amazon WS	EUCALYPTUS
Indicator	Peak-time	Capacity VM Instance	Request processing time
Complexity	Low	High	High

3. Proposed Architecture

Resource Monitoring

Resource monitoring module is a module to continuously collect workload information about the use of hardware (CPU, memory, disk, and network) and software (file handles and modules). This module then uses this information for the future decision to allocate demanded resource.

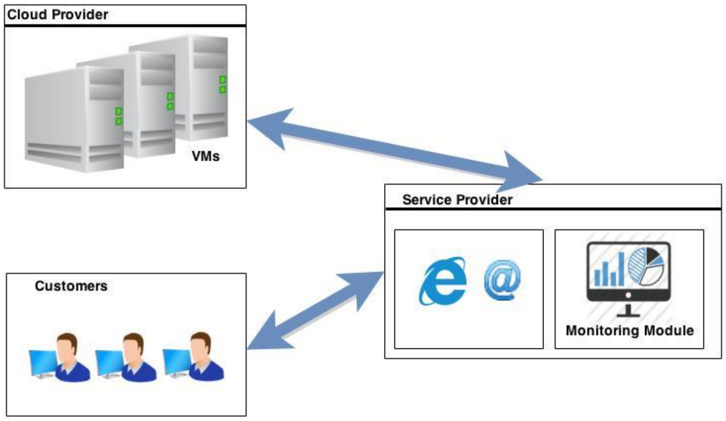

Figure 1. General Architecture

Scenario Used

The scenario used in this architecture is described as follow. There are 3 main actors involved in this scenario; Cloud provider as an infrastructure provider (VMs), the service provider (SP) who provides internet services, and users accessing SP’s services. All actors are connected each other via an internet connection. Main focus observed in this paper is the service provider. In figure 1, service provider leases virtual resources (VMs) from the cloud provider. Hence, service provider deploys their services such as website, email service, or other software applications on those VMs. These services will be accessed by users through an internet connection.

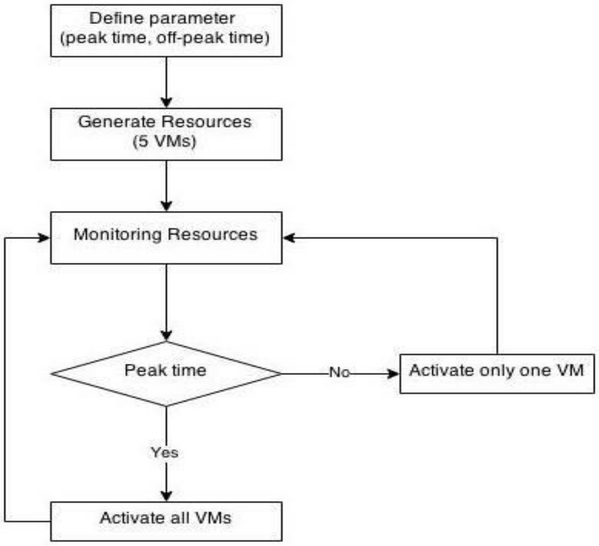

Figure 2. Flow Chart

To maintain its services to the users, it is important for the service provider to monitor their resources. This monitoring module will monitor VMs workload which is hosted in cloud provider infrastructure. Therefore service provider can use this resource monitoring module to monitor current workload, then uses scheduled-based approach to allocating resources needed to serve users.

4. Evaluation and Discussion

To evaluate this approach, we built the component and network based on the architecture in figure 1

Figure 1. In addition, we divided into two main important parts; cloud provider side to host our virtual machines, and service provider side to apply monitoring module.

- Cloud provider side: we use *Google Compute Engine (GCE)* as an IaaS model to host our VMs. Google Compute Engine (GCE) is the Infrastructure as a Service (IaaS) component of Google Cloud Platform which is built on the global infrastructure that runs Google's search engine, Gmail, YouTube and other services. Google Compute Engine enables users to launch virtual machines (VMs) on demand. To evaluate our approach, we installed a web server on the VMs. This web server then will handle user request (HTTP request connection), and later we observe network traffic and VM resources load.
- Service provider side: on this side, a resource monitoring module is used to observe VMs workload. We use Locust which is python based code as a testing tool to generate user request. It is completely event based, therefore it is possible to support thousands of concurrent users on a single machine [8]. The scenario will be defined as the following figure.

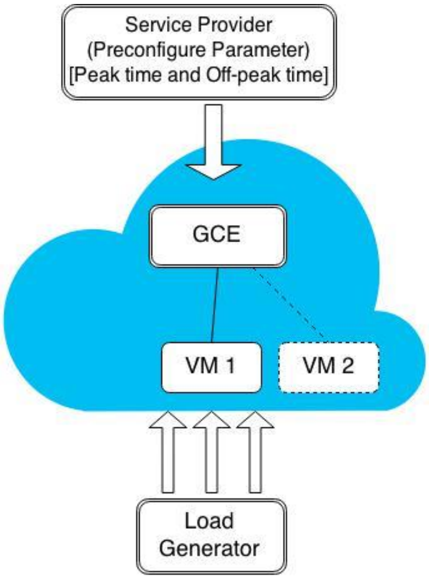

Figure 3. Evaluation Scenario

In this paper, we define peak time and off-peak time as follow: (a) peak time: 8.00 AM – 5.00 PM and (b) off-peak time: 5.00 PM – 08.00 AM. The evaluation scenario will monitor the workload of VMs hosted in GCE. During the peak time, we will generate high workload traffic and assign all the VMs to satisfy user request. On the other hand, during the off-peak time we only assign one VM to satisfy the user request and generate low workload traffic.

a. Experimental Setup

To simulate our approach, we use the following parameters which are shown in Table 2.

Table 2. Experimental Setup

Items	Description
VM	Google Cloud Engine [6]
VM Specs	N1-standard-1 (1 vCPU), 3.8 GB RAM, 10 GB HDD
Testing Tool	Locust 0.7.2 python-based [8]
Testing Parameter:	
Peak Time	1000, 2000, 3000 users 1 to 4 VMs + 1 distribution VM 10 Req / sec
Off-Peak Time	500 users 1 VM 10 Req / sec

b. Simulation Results

In this experiment, we measure some parameters such as resource utilization, successful requests and failure rate with a different number of user requests for a different number of active Vms.

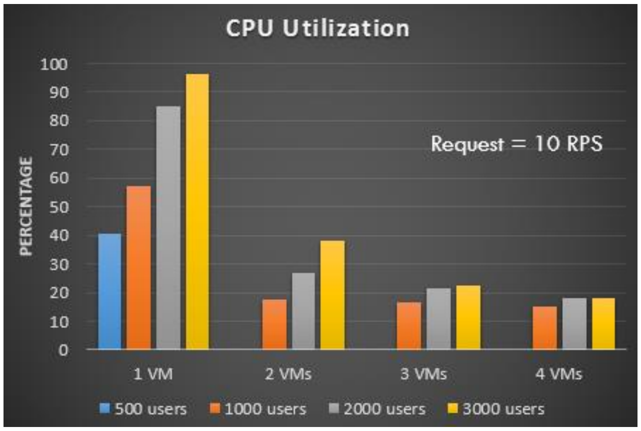

Figure 4. CPU Utilization

The first experiment is to measure CPU utilization. Figure 4 shows that more resources are needed during peak time (represented by a large number of requests). Not only CPU resources are needed, but also it depends on the network bandwidth. Therefore more requests must be handled by activating more VMs.

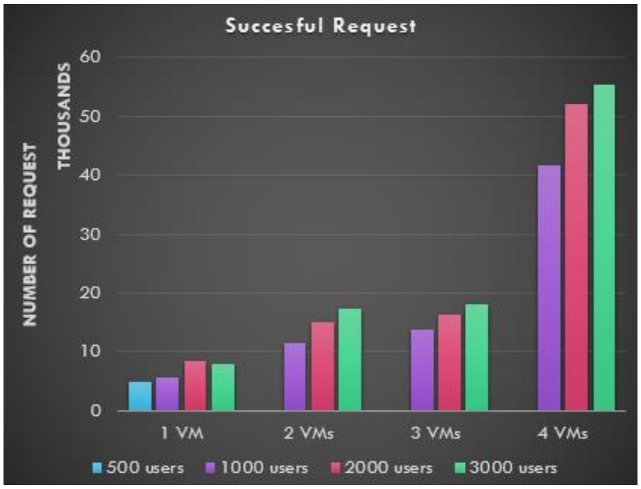

Figure 5. Successful Request

The second experiment is shown in figure 5. Sending more requests cannot be handled by only 1 VM due to the limited resources. Therefore the more the number of the requests, the more VMs are needed. From the result, it can be seen that successful requests sent depend on the amount of active VMs which handles the request.

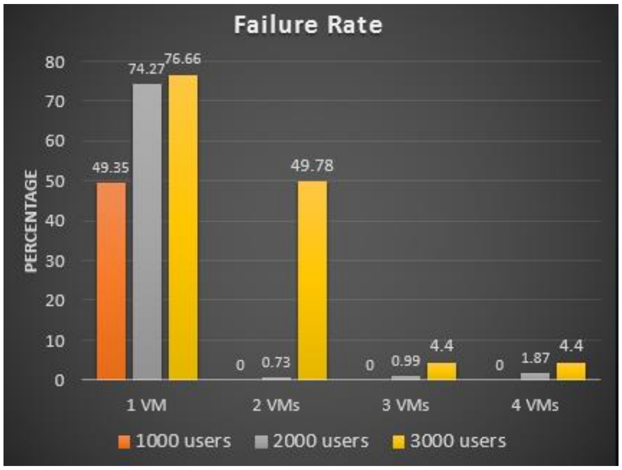

Figure 6. Request Failure Rate

The relation with figure 5 can be seen in figure 6 . It shows the failure rate of the requested sent to the VMs. Failure rate can be reduced by allocating more VMs to satisfy the huge number of requests. As we can see in the picture, we need more VMs as the number of requests grows to achieve zero failure.

5. Conclusion and Future Work

In this paper, we proposed a monitoring and resource allocating module for service providers to maintain their VMs hosted on the VMs cloud server. It is important to observe

VMs workload before publicly releasing it to the cloud, so the service provider can estimate the resources needed in the peak or off-peak time in order to satisfy user requests with optimal resources utilization. To evaluate our approach, we use Google Compute Engine as VM cloud server and Locust as a monitoring module, then statically allocate appropriate resources based on scheduled time. The results from these experiments show that the higher the number of the requests, the more resources needed. More allocated resources are able to handle user requests with minimum failure rate.

The proposed approach only can monitor VMs workload based on certain given time, but this metric is very limited. In order to dynamically adapt with real-time traffic condition, it is needed to add more flexible metrics. Therefore the future direction of this paper will be the expansion of scaling method by applying dynamic resource allocation.

References

[1] Ming Mao, Humphrey M., “Scaling and Scheduling to Maximize Application Performance within Budget Constraints in Cloud Workflow,” in proc. IEEE 27th International Symposium on IPDPS, pp. 67-78, Boston , 20-24 May 2013.

[2] W. Iqbal, et al., “SLA-Driven Adaptive Resource Management for Web Applications on a Heterogeneous Compute Cloud,” in proc. *CloudCom ’09 Proceeding of the 1st International Conference on Cloud Computing*, pp. 243–253, Springer-verlag Berlin, Heidelberg, 2009.

[3] Weifan Hong, et al., “Application-aware Resource Allocation for SDN-based Cloud Data Center”, in proc. IEEE International Conference on Cloud Computing and Big Data (CloudCom-Asia), pp. 106-110, 16-19 December 2013

[4] Y. Lee, C. Wang, A. Zomaya and B. Zhou. 2010., “Profit-driven service request scheduling in clouds”, In 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing, Melbourne, Victoria, Australia, May 17-20, 2010)

[5] M. Mao, J. Li and M. Humphrey. 2010. “Cloud auto-Scaling with deadline and budget constraints”, In Proceedings of 11th ACM/IEEE International Conference on Grid Computing, Brussels, Belgium, Oct 25-28, 2010.

[6] Google Cloud Platform [online], Available: https://cloud.google.com.

[7] Apache JMeter Testing [online], Available: http://jmeter.apache.org.

[8] An Open Source Load Testing Tool, [online], Available: http://locust.io.