Amazon Elastic Beanstalk

SHREE GOVIND MISHRA[1]

[1] *School of Informatics and Computing, Bloomington, IN 47408, U.S.A.*
[*] *Corresponding authors: shremish@indiana.edu*

project-000, April 1, 2017

Amazon Elastic Beanstalk is a service provided by the Amazon Web Services which allow developers and engineers to easily deploy and run web applications in the cloud, in such a way that these applications are highly available and scalable. Elastic Beanstalk manages the deployed application by reducing the management complexities as it automatically handles the capacity provisioning, load balancing, scaling, and application health monitoring. Elastic BeanStalk also provisions one or more AWS Resources such as Amazon EC2 instances when an application is deployed.

Keywords: Cloud, I524, Amazon Web Services, AWS Elastic Beanstalk, Load Balancing, Cloud Watch, AWS Management Console, Auto Scaling

https://github.com/cloudmesh/sp17-i524/raw/master/paper2/S17-IR-2021/report.pdf

INTRODUCTION

Cloud Computing can be defined as an abstraction of services from infrastructures(i.e hardware), platforms and applications(i.e software) by virtualization of resources [1]. The different form of cloud computing services include IaaS, PaaS, and SaaS which stands for (Infrastructure as a Service), Platform as a Service, and Software as a service respectively. With Cloud Computing, organizations can consume shared computing and storage resources rather than building, operating and improving infrastructure on their own and it can also enable organizations to obtain a flexible, secure and cost-effective Infrastructure.

Amazon Web Services (AWS) is a subsidiary of Amazon.com, which offers a suite of cloud computing services such as compute power, database storage, content delivery and other functionalities that make up an on-demand computing platform. The scaling on IaaS level can be illustrated with an example of Amazon Elastic Compute Cloud(AmazonEC2), whereas scaling on PaaS level can be illustrated with Amazon Web Services Elastic Beanstalk(AWS Elastic Beanstalk). AWS Elastic Beanstalk is one of the many services provided by the AWS with its functionality to manage the Infrastructure.Elastic beanstalk provides a quick deployment and management of the applications on the Cloud as it automatically handles the details of capacity provisioning, load balancing, scaling, and application health monitoring. Elastic beanstalk uses highly reliable and scalable services [2].

NEED FOR AWS ELASTIC BEANSTALK

Cloud Computing shifts the location of resources to the cloud to reduce the cost associated with over-provisioning, under-provisioning, and under-utilization. When more than required resources are available, is called over-provisioning [3]. When the resources are not used adequately, it is known as under-utilization. And when the resources available are not enough is called under-provisioning. Cloud Computing should also reduce the time required to provision the resources with the variable workload on the server so that the applications can be quickly scaled up and down with the variable workload [4]. There scaling of applications can be achieved by:

Manual Scaling in Cloud Enviornment

In traditional applications, scalability is achieved by predicting the peak loads, then purchasing, setting up and configuring the infrastructure that could handle this peak load [5]. With Manual Scaling, the resources are provisioned at the deployment time and the application servers are added to infrastructure manually, thus there is high latency. Due to these issues, the average time the applications are provisioned is long. There is also an issue of Manual Monitoring of the resources allocated.

Semi-Manual Scaling in Cloud Enviornment

The resources provided for the Inftrastruare are virtualized in the cloud environment and thus this virtualization enables the elasticity. In particular, in cloud environments, resources are provisioned dynamically (i.e. at runtime), automatically (i.e. without user intervention), infinitely and almost immediately (i.e. within minutes and not hours, days, weeks or months like in traditional environments) [1]. Thus, the mean time until when the resources are provisioned are short. However, manual monitoring of resources is still necessary. In particular, users are forced to make a tradeoff between requesting more resources to avoid under-provisioning and requesting fewer resources to

avoid over-provisioning and under-utilization. Since resources are provisioned by request, the problem of the unavailability of an application at peak loads is not completely eliminated.

Automatic Scaling in Cloud Enviornment

Automatic Scaling enables users to closely follow the workload curve of their application, by provisioning resources on demand. The user owns the choice that the number of resources these applications are using increases automatically during the time when the demand of resources are high to handle the peak load and also automatically decreases when the minimal resources are needed, to minimize the cost so that the user only pays for what they used. Automatic Scaling also predicts the peak load that the applications may require in the future and provisions these required resources in advance, proving the elasticity of the cloud [1].

ELASTIC BEANSTALK SERVICES

Elastic Beanstalk supports applications developed in Java, PHP, NET, Node.js, Python, and Ruby, and also in different container types for each language. A container defines the infrastructure and software stack to be used for a given environment. When an application is deployed, Elastic Beanstalk provisions one or moreAWS resources, such as Amazon EC2 Instances [4]. The software stack that runs on the instances depends on the container type, where two container types are supported by the Elastic Beanstalk Node.js: a 32-bit Amazon Linux Image and a 64-bit Amazon Linux Image. Where each of them runs the Software stack tailored to the hosted Node.js application. Amazon Elastic Beanstalk can be interacted using the AWS Management Console, the AWS Command Line Interface(AWS CLI), or a high-level CLI designed for Elastic BeanStalk [4].

Amazon Elastic Beanstalk provides the Automatic Scaling in cloud Environments. For Automatic Scaling, it uses the following Amazon Web Services:

AWS Management Console

AWS Management Console is a browser-based graphical user interface(GUI) for Amazon Web Services. It allows users to configure an automatic scaling mechanism of AWS Elastic Beanstalk as well as other services of AWS. From the Management console, the user can decide about how many instances does the application require. When the application must be scaled up and down [1].

Elastic Load Balancing

Elastic Load Balancing enables the load balancer which automatically distributes the incoming application traffic across all running instances in the auto-scaling group based on metrics like request count and latency tracked by Amazon Cloud Watch. If an instance is terminated, the load balancer will not route requests to this instance anymore. Rather, it will distribute the requests across the remaining instances.

Elastic Load Balancing also monitors the availability of an application, by checking its "health" periodically (e.g. every five minutes). If this check fails, AWS Elastic Beanstalk will execute further tests to detect the cause of the failure [6]. In particular, it checks if the load balancer and the auto-scaling group are existing. In addition, it checks if at least one instance is running in the auto-scaling group. Depending on the test results, AWS Elastic Beanstalk changes the health status of the application.

The different colors respond to the status of the application. Green indicates that the application responded within a minute, Yellow indicates that the application has not responded in the last 5 minutes, Red indicates that application has not responded in the last 5 minutes or some other problem may have been detected by AWS Elastic Beanstalk. Gray indicates that the status of the application is unknown as of now [6].

Auto Scaling

Auto Scaling automatically launches and terminates instances based on metrics like CPU and RAM utilization of the application and are tracked by Amazon CloudWatch and thresholds called triggers. Whenever a metric crosses a threshold, a trigger is fired to initiate automatic scaling. For example, a new instance will be launched and registered at the load balancer if the average CPU utilization of all running instances exceeds an upper threshold

Auto Scaling also provides fault tolerance. If an instance reaches an unhealthy status or terminates unexpectedly, Auto Scaling will compensate this and launch a new instance instead, thus assuring that the specified minimum number of instances are running constantly.

Amazon Cloud Watch

Amazon CloudWatch is a component of Amazon Web Services (AWS) that provides monitoring for AWS resources and the customer applications running on the Amazon infrastructure [7] It tracks and stores per-instance metrics, including request count and latency, CPU and RAM utilization. Once stored, the metrics can be visualized an API, command-line tools, one of the AWS SDK (software development kits) or the AWS Management Console. With AWS Management Console, users can get real-time visibility into the utilization of each of the instances in an auto-scaling group via the graphs and chart provided, and can easily and quickly detect the over-provisioning, under-utilization or under-provisioning.

CONCLUSION

In traditional (i.e. non-cloud) environments, overprovisioning and under-utilization can hardly be avoided [**?**]. There is an observation that in many companies the average utilization of application servers ranges from 5 to 20 percent, meaning that many resources like CPU and RAM are idle at peak times [8]. On the other hand, if the companies shrink their infrastructures to reduce over-provisioning and under-utilization, the risk of under-provisioning will increase. While the costs of over-provisioning and under-utilization can easily be calculated, the costs of under-provisioning are more difficult to calculate because under-provisioning can lead to a no peak times [8].

Other platforms like Google App Engine [9] also let users have their applications to automatically scale both up and down according to demand but with even more restrictions on how users should develop their applications. Wheras, with AWS Elastic Beanstalk there is more control with the user. A downside of AWS Elastic Beanstalk is that currently, it does not provide any web service to predict demand for the near future. Theoretically, the statistical usage data of the last few months or years could be used to predict time intervals during which more or fewer resources are needed.

REFERENCES

[1] D. Bellenger, J. Bertram, A. Budina, A. Koschel, B. Pfänder, C. Serowy, I. Astrova, S. G. Grivas, and M. Schaaf, "Scaling in cloud environments," *ACM Digital Library*, pp. 145–150, 2011. [Online]. Available: http://www.wseas.us/e-library/conferences/2011/Corfu/COMPUTERS/COMPUTERS-23.pdf

[2] Amazon Web Services, "Waht is aws elastic beanstalk," Web page, 2017, online; accessed 19-Mar-2017. [Online]. Available: http://docs.aws.amazon.com/elasticbeanstalk/latest/dg/Welcome.html

[3] S. Tai, J. Nimis, C. Baun, and M. Kunze, *Cloud Computing: Web-Based Dynamic IT Services 1st*. Springer Publishing Company, Incorporated ©2011. [Online]. Available: http://www.springer.com/us/book/9783642209161

[4] J. Vlie, F. Paganelli, S. van Wel, and D. Dowd, *Elastic Beanstalk*, 1st ed. O'Reilly Media, Inc., 2011, online; accessed 18-Mar-2017.

[5] J. Yang, J. Qiu, and Y. Li, "A profile-based approach to just-in-time scalability for cloud applications." IEEE Computer Society Washington, DC, USA ©2009, 2009, pp. 9–16.

[6] Amazon Web Services, "Elastic load balancing," Web page, online; accessed 20-Mar-2017. [Online]. Available: https://aws.amazon.com/elasticloadbalancing/

[7] TechTarget, "Searchaws CloudWatch," Web page, online; accessed 21-Mar-2017. [Online]. Available: http://searchaws.techtarget.com/definition/CloudWatch

[8] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "Communications of the acm," *ACM Digital Library*, vol. 53, pp. 50–58. [Online]. Available: http://cacm.acm.org/magazines/2010/4/81493-a-view-of-cloud-computing/fulltext

[9] Google Inc., "Google cloud platformGoogle App Engine," Web page, online; accessed 18-Mar-2017. [Online]. Available: https://cloud.google.com/appengine/