A PROJECT REPORT
ON

"VIRTUAL MACHINE SCALING"

Based on performance fluctuation in public cloud

Bachelor of Technology
Computer Science Engineering

BY

Ayush Sharma	2016KUCP1027
Devesh Gaur	2016KUCP1029
Rajesh Kumar	2016KUCP1040

UNDER THE GUIDANCE OF

Dr. Isha Pathak Tripathi

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING

INDIAN INSTITUTE OF INFORMATION TECHNOLOGY, KOTA

2016-2020

DECLARATION

I hereby declare that the work reported in the B.Tech report entitled "Virtual Machine Scaling " submitted at Indian Institute of Information Technology , Kota is an authentic record of our carried work under the supervision of Dr. Isha Pathak Tripathi. I have not submitted this work elsewhere for this degree.

Ayush Sharma
2016kucp1027

Devesh Gaur
2016kucp1029

Rajesh Kumar
2016kucp1040

Under the supervision of
Dr. Isha Pathak Tripathi

Department of Computer Science Engineering
Indian Institute of Information Technology,Kota

Acknowledgements

We have taken efforts in this project. However, it would not have been possible without the kind support and help of many individuals and organizations. We would like to extend our sincere thanks to all of them.

We are highly indebted to **Dr. Isha Pathak Tripathi** for their guidance and constant supervision as well as for providing necessary information regarding the project also for their support in completing the project. We would like to express my gratitude towards our parents friends for their kind co-operation and encouragement which helped us in completion of this project. We would like to express our special gratitude and thanks to industry persons for giving us such attention and time. Our thanks and appreciations also go to our colleague in developing the project and people who have willingly helped us out with their abilities.

Ayush Sharma
2016kucp1027

Devesh Gaur
2016kucp1029

Rajesh Kumar
2016kucp1040

ABSTRACT

Cloud computing has been adapted for various application areas because it can reduce the time required for system development and the cost of hardware. One of the factors that degrades performance stability of applications running in the cloud is unexpected loads, caused by interference between Virtual Machines (VMs) co-existing on the same physical machine. In this paper, we propose a VM scaling method that forecasts performance fluctuation caused by unexpected loads to adjust the number of VMs appropriately.

Although the investment in Cloud Computing incredibly grows in the last few years, the offered technologies for dynamic scaling in Cloud Systems don't satisfy, neither nowadays fluky applications (i.e. social networks, web hosting, content delivery) that exploit the power of the Cloud, nor the energy challenges caused by its datacentres.

Hence we propose a proactive model based on an application behaviours, prediction technique to predict the future workload behaviour of the virtual machines (VMs) executed at Cloud hosts. The predicted information can help VMs to dynamically and proactively be adapted to satisfy the provider demands in terms of increasing the utilization and decreasing the power consumption, and to enhance the services in terms of improving the performance with respect to the Quality of Services (QoS) requirements and dynamic changes demands.

Contents

List of Figures

Chapter 1

Introduction

AWS virtual machine scale sets let you create and manage a group of identical, load balanced VMs. The number of VM instances can automatically increase or decrease in response to demand or a defined schedule. Scale sets provide high availability to your applications, and allow you to centrally manage, configure, and update a large number of VMs. With virtual machine scale sets, you can build large-scale services for areas such as compute, big data, and container workloads.

1.1 Why use virtual machine scale sets?

To provide redundancy and improved performance, applications are typically distributed across multiple instances. Customers may access your application through a load balancer that distributes requests to one of the application instances. If you need to perform maintenance or update an application instance, your customers must be distributed to another available application instance. To keep up with additional customer demand, you may need to increase the number of application instances that run your application.

1.1.1 Easy to create and manage multiple VMs

When you have many VMs that run your application, it's important to maintain a consistent configuration across your environment. For reliable performance of your application, the VM size, disk configuration, and application installs should match across all VMs. With scale sets, all VM instances are created from the same base OS image and configuration. This approach lets you easily manage hundreds of VMs without additional configuration tasks or network management.

1.1.2 Provides high availability and application resiliency

Scale sets are used to run multiple instances of your application. If one of these VM instances has a problem, customers continue to access your application through one of the other VM instances with minimal interruption. For additional availability, you can use Availability Zones to automatically distribute VM instances in a scale set within a single datacenter or across multiple datacenters.

1.1.3 Allows your application to automatically scale as resource demand changes

Customer demand for your application may change throughout the day or week. To match customer demand, scale sets can automatically increase the number of VM instances as application demand increases, then reduce the number of VM instances as demand decreases. Autoscale also minimizes the number of unnecessary VM instances that run your application when demand is low, while customers continue to receive an acceptable level of performance as demand grows and additional VM instances are automatically added.

1.1.4 Works at large-scale

Scale sets support up to 1,000 VM instances. If you create and upload your own custom VM images, the limit is 600 VM instances.

Chapter 2

Literature Survey

2.1 Literature Survey

Authors: Yu Kaneko, Toshio Ito, Masashi Ito, Hiroshi Kawazoe

2017 IEEE 10th International Conference on Cloud Computing Engineering -Publiser: IEEE

Cloud computing has been adapted for various application areas.Users of public cloud can use Virtual Machines (VMs) as a runtime environment for applications, which leads to reduction of development time and hardware cost because users do not need to procure any hardware. Some research projects are ongoing to migrate Industrial Control Systems (ICS) systems to a cloud environment. ICS require high stability of performance to monitor and control devices accurately. Public cloud providers typically guarantee a certain level of availability of VMs, but normally do not guarantee performance stability of VMs. Achieving high stability of performance in the public cloud is users responsibility and is a challenge when migrating ICS to a cloud environment. This paper proposes a VM scaling method that forecasts expected performance change and unexpected performance fluctuation separately.

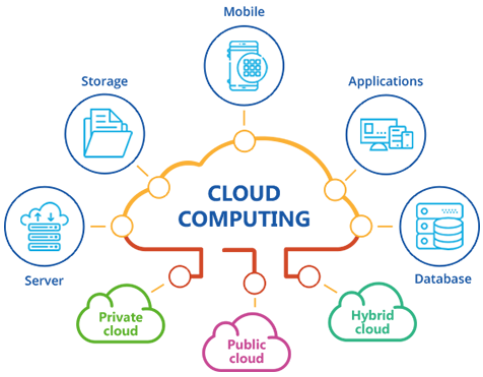

Figure 2.1: Cloud Computing Services

Chapter 3

Understanding Cloud Computing(AWS)

Cloud computing is named as such because the information being accessed is found remotely in the cloud or a virtual space. Companies that provide cloud services enable users to store files and applications on remote servers and then access all the data via the internet. This means the user is not required to be in a specific place to gain access to it, allowing the user to work remotely.

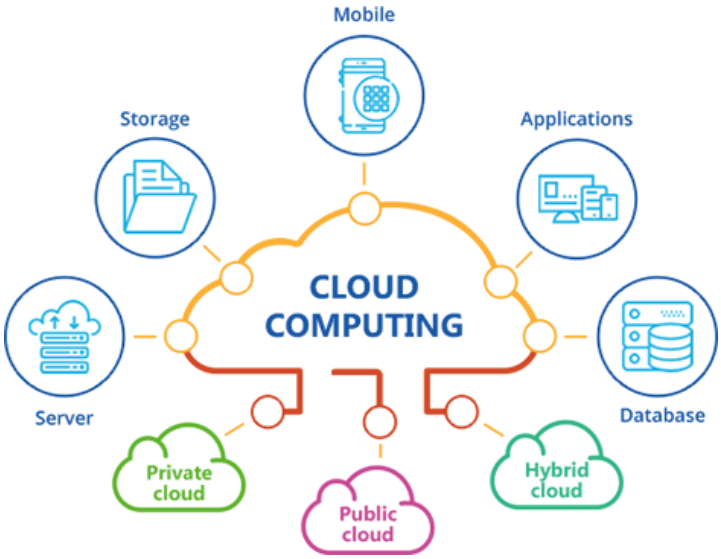

Figure 3.1: Cloud Computing Services

Cloud computing takes all the heavy lifting involved in crunching and processing data away from the device you carry around or sit and work at. It also moves all of that work to huge computer clusters far away in cyberspace. The internet becomes the cloud, and voilyour data, work, and applications are available from any device with which you can connect to the internet, anywhere in the world.

Figure 3.2: Public Cloud Providers

Cloud computing can be both public and private. Public cloud services provide their services over the internet for a fee. Private cloud services, on the other hand, only provide services to a certain number of people. These services are a system of networks that supply hosted services. There is also a hybrid option, which combines elements of both the public and private services. More information is outlined below.

Cloud Computing Deployment Models

There are various types of clouds, each of which is different from the other.

Public clouds provide their services on servers and storage on the internet. These are operated by third-party companies, who handle and control all the hardware, software, and the general infrastructure. Clients access services through accounts which can be accessed by just about anyone.

Private clouds are reserved for specific clientele, usually one business or organization. The firm's data service center may host the cloud computing service. Many private cloud computing services are provided on a private network.

Hybrid clouds are, as the name implies, a combination of both public and private services. This type of model allows the user more flexibility and helps optimize the user's infrastructure and security.

Security

Many cloud providers offer a broad set of policies, technologies and controls that strengthen your security posture overall, helping protect your data, apps and infrastructure from potential threats.

3.1 AWS

Amazon Web Services (AWS) is a secure cloud services platform, offering compute power, database storage, content delivery and other functionality to help businesses scale and grow.

AWS launched in 2006 from the internal infrastructure that Amazon.com built to handle its online retail operations. AWS was one of the first companies to introduce a pay-as-you-go cloud computing model that scales to provide users with compute, storage or throughput as needed.

Amazon Web Services provides services from dozens of data centers spread across availability zones (AZs) in regions across the world. An AZ represents a location that typically contains multiple physical data centers, while a region is a collection of AZs in geographic proximity connected by low-latency network links. An AWS customer can spin up virtual machines (VMs) and replicate data in different AZs to achieve a highly reliable infrastructure that is resistant to failures of individual servers or an entire data center.

More than 100 services comprise the Amazon Web Services portfolio, including those for compute, databases, infrastructure management, application development and security.

3.2 AWS Developer tools Used in our project:

3.2.1 EC2 Service

Amazon Elastic Compute Cloud (EC2) provides virtual servers – called instances – for compute capacity. The EC2 service offers dozens of instance types with varying capacities and sizes, tailored to specific workload types and applications, such as memory-intensive and accelerated-computing jobs. AWS also provides an Auto Scaling tool to dynamically scale capacity to maintain instance health and performance.

The Amazon EC2 Container Service and EC2 Container Registry enable customers to work with Docker containers and images on the AWS platform. A developer can also use AWS Lambda for serverless functions that automatically run code

for applications and services, as well as AWS Elastic Beanstalk for PaaS. AWS also includes Amazon Lightsail, which provides virtual private servers, and AWS Batch, which processes a series of jobs.

3.2.2 Load Balancer

Elastic Load Balancing distributes incoming application or network traffic across multiple targets, such as Amazon EC2 instances, containers, and IP addresses, in multiple Availability Zones. Elastic Load Balancing scales your load balancer as traffic to your application changes over time, and can scale to the vast majority of workloads automatically.

Load Balancer Benefits

A load balancer distributes workloads across multiple compute resources, such as virtual servers. Using a load balancer increases the availability and fault tolerance of your applications.

You can add and remove compute resources from your load balancer as your needs change, without disrupting the overall flow of requests to your applications.

You can configure health checks, which are used to monitor the health of the compute resources so that the load balancer can send requests only to the healthy ones. You can also offload the work of encryption and decryption to your load balancer so that your compute resources can focus on their main work.

Features of Elastic Load Balancing

Elastic Load Balancing supports three types of load balancers: Application Load Balancers, Network Load Balancers, and Classic Load Balancers. You can select a load balancer based on your application needs. For more information, see Comparison of Elastic Load Balancing Products.

For more information about using each load balancer, see the User Guide for Application Load Balancers, the User Guide for Network Load Balancers, and the User Guide for Classic Load Balancers.

3.2.3 CloudWatch

Amazon CloudWatch is a monitoring and management service built for developers, system operators, site reliability engineers (SRE), and IT managers. CloudWatch provides you with data and actionable insights to monitor your applications, understand and respond to system-wide performance changes, optimize resource utilization, and get a unified view of operational health. CloudWatch collects monitoring

and operational data in the form of logs, metrics, and events, providing you with a unified view of AWS resources, applications and services that run on AWS, and on-premises servers. You can use CloudWatch to set high resolution alarms, visualize logs and metrics side by side, take automated actions, troubleshoot issues, and discover insights to optimize your applications, and ensure they are running smoothly.

With Amazon CloudWatch, it is easy to get started. There is no up-front commitment or minimum fee; you simply pay for what you use. You will be charged at the end of the month for what you use.

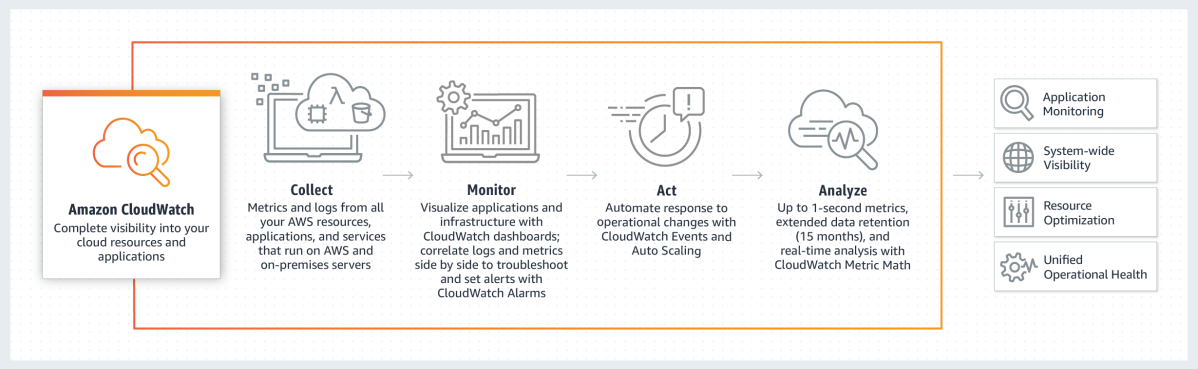

Figure 3.3: How Cloudwatch works

Chapter 4

System Architecture

Fig. 1 shows the system architecture assumed in this paper. The system consists of a load balancer and VMs and each VM runs. Http Requests will be given to the load balancer through other virtual machine dedicated for sending requests. The load balancer distributes received messages to the VMs. The VM manager measures the number of incoming messages as expected loads and the cpu utilization of each virtual machine as its performance. Then, it forecasts expected performance changes and unexpected performance fluctuation to determine the appropriate number of VMs.

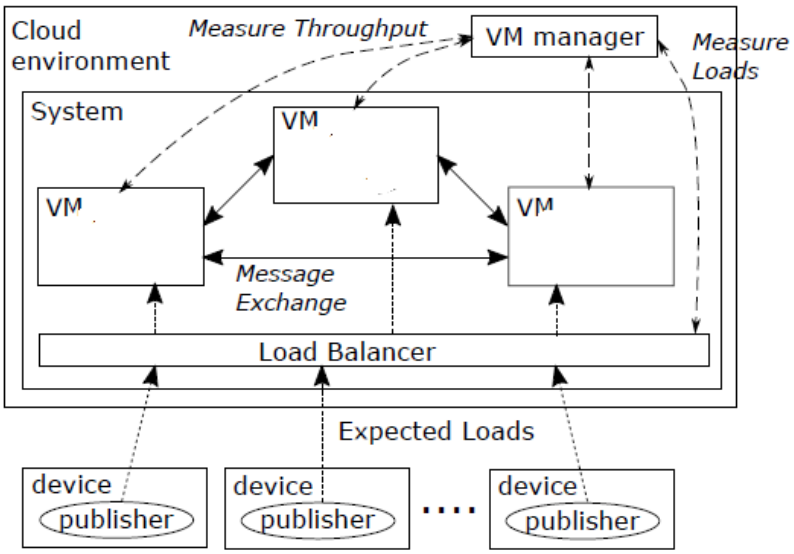

Fig. 1. System architecture assumed in this paper

Chapter 5

Proposed Method

5.1 Formulation between Expected Loads and Performance

To detect unexpected performance fluctuation, we formulate a relationship between expected loads and performance of a VM in an environment where no unexpected loads exist. The relationship can be figured out based on a simple performance evaluation in a private environment. Multiple pairs of an expected load and performance can be obtained from measurements with various amounts of expected loads. A relationship between an expected load L and performance P can be formulated by applying a fitting method such as regression analysis for the pairs (1).

$$P = f(L) \quad \text{..... (1)}$$

5.2 Forecast of Unexpected Performance Fluctuation and Expected Loads

Amount of unexpected performance fluctuation of a VMi at time t with an expected load Lt,i is calculated by (2).
Lt,i is an expected load distributed for VMi at t by the load balancer.

$$A_{t,i} = f(L_{t,i}) \; P_{t,i} \quad \text{.....(2)}$$

f(Lt,i) denotes that performance of VMi is affected only by an expected load Lt,i.
Pt,i denotes that measured (actual) performance of VMi at t.

Therefore, the difference between f(Lt,i) and Pt,i can be considered to be an amount of unexpected performance fluctuation. The VM manager records At,i as time series

data. Future unexpected performance fluctuation of VMi can be forecast by (3).

$$A_{t+1,i} = \text{forecast}(A_{t,i}, A_{t1,i}, ..., A_{tK+1,i}) \quad .....(3)$$

Equation 3 forecasts an amount of unexpected performance fluctuation of VMi at (t+1) from K latest time series data. The proposed method ignores the detail of the forecast method. For example, ExponentialWeighted Moving Average (EWMA) can be used as the forecast method.

The VM manager also measures expected loads for the load balancer and records them as time series data to forecast future expected loads. A future expected load at (t + 1) is forecast based on J latest measured expected loads by (4).

$$L_{t+1} = \text{forecast}(L_t, L_{t1}, ..., L_{tJ+1}) \quad .....(4)$$

5.3 Forecast of VM Performance

The VM manager forecasts performance of VMs based on forecasts of expected loads and unexpected performance fluctuation. Hereafter, we refer to current time as t and future time as (t + 1).
M denotes the number of VMs at t, and N denotes the number of VMs at (t + 1).
First, the VM manager forecasts total amount of expected load L_{t+1} by (4). Then, it calculates future expected loads for each VM ($L_{t+1,1}$, ..., $L_{t+1,N}$) based on the distribution algorithm of the load balancer. For example, if the load balancer distributes an expected load for VMs equally, $L_{t+1,i} = L_{t+1}$ N . The VM manager also forecasts unexpected performance fluctuation of each VM ($A_{t+1,1}$, ..., $A_{t+1,N}$) by (3). The VM manager does not have time series data of performance fluctuation of VMi ($i > M$), and therefore it uses $A_{t+1,M}$ as $A_{t+1,i}$ ($i > M$). The VM manager finally calculates future performance $P_{t+1,i}$ of VMi, considering unexpected performance fluctuation by (5)

$$P_{t+1,i} = f(L_{t+1,i}) \; A_{t+1,i} \quad .....(5)$$

5.4 Algorithm Used

The VM manager calculates the appropriate number of VMs according to future performance of VMs forecast by (5). The adjustment algorithm is as follows.

$N \leftarrow 1$
Forecast: L_{t+1}
loop
  Forecast: $L_{t+1,1}, L_{t+1,2}, ..., L_{t+1,N}$
  Forecast: $A_{t+1,1}, A_{t+1,2}, ..., A_{t+1,N}$
  Forecast: $P_{t+1,1}, P_{t+1,2}, ..., P_{t+1,N}$
  $AllGreen \leftarrow true$
  for $i = 1$ to N **do**
    if $P_{t+1,i}$ does not satisfy TP **then**
      $AllGreen \leftarrow false$
    end if
  end for
  if $AllGreen$ is $true$ **then**
    break
  else
    $N \leftarrow N + 1$
  end if
end loop
if $N > M$ **then**
  Create $(N - M)$ VMs
else if $N < M$ **then**
  Stop $(M - N)$ VMs
end if

Chapter 6

Results and Findings

6.1 Case 1(Overutilization): Creation of new virtual machine if target performance not satisfied

1. At first a single Vitual Machine is put to test.

2. Around 20000 http requests per 5 min are given from a virtual machine in the same region where the VM on test is available. Region is kept same so as to avoid latency.

3. Accordingly CPU utilization of the test VM is recorded for atleast 3hrs continuously.

4. If CPU utilization is more(say 90+) the performance of the VM is affected. So, on exeecding the recommended target performance, a new VM is created and registered with the Load Balancer so as to divide the load between the old VM and the newly created VM.

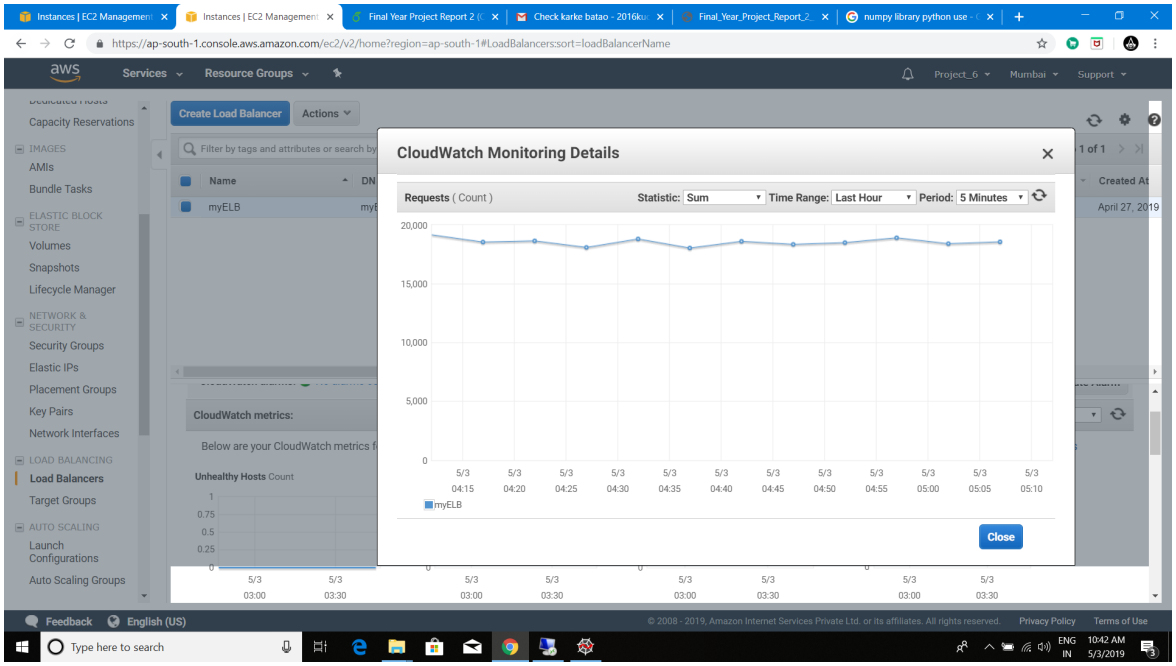

Figure 6.1: Fig. Number of Http Requests

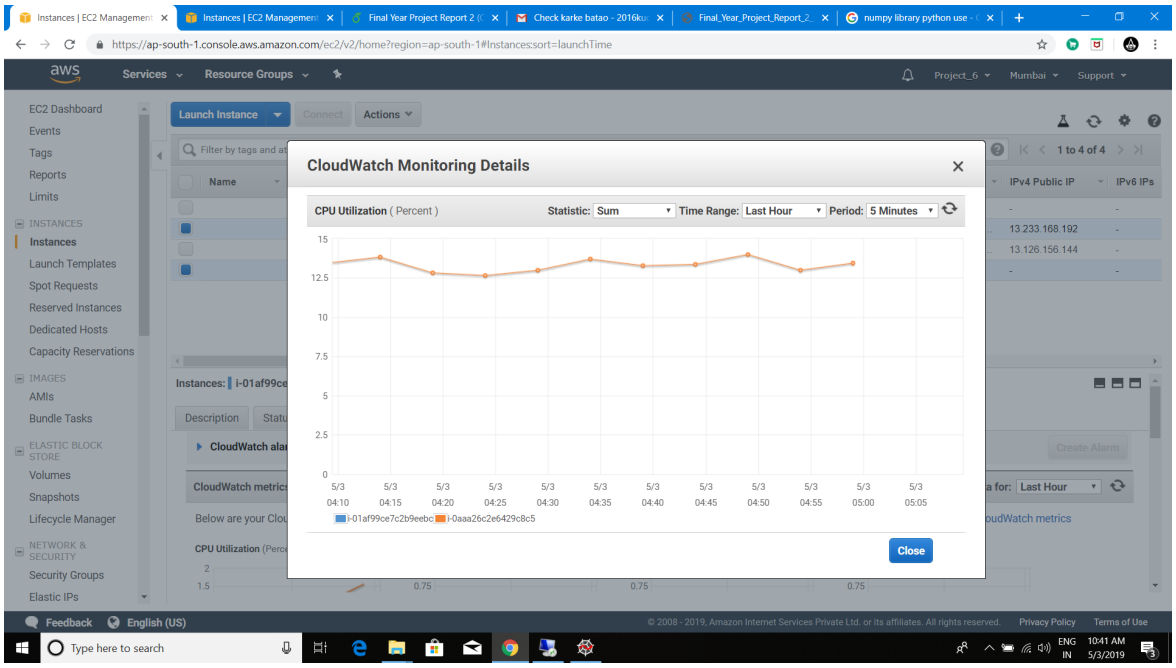

Figure 6.2: Fig. CPU Utilization of Running VM exceeding target performance

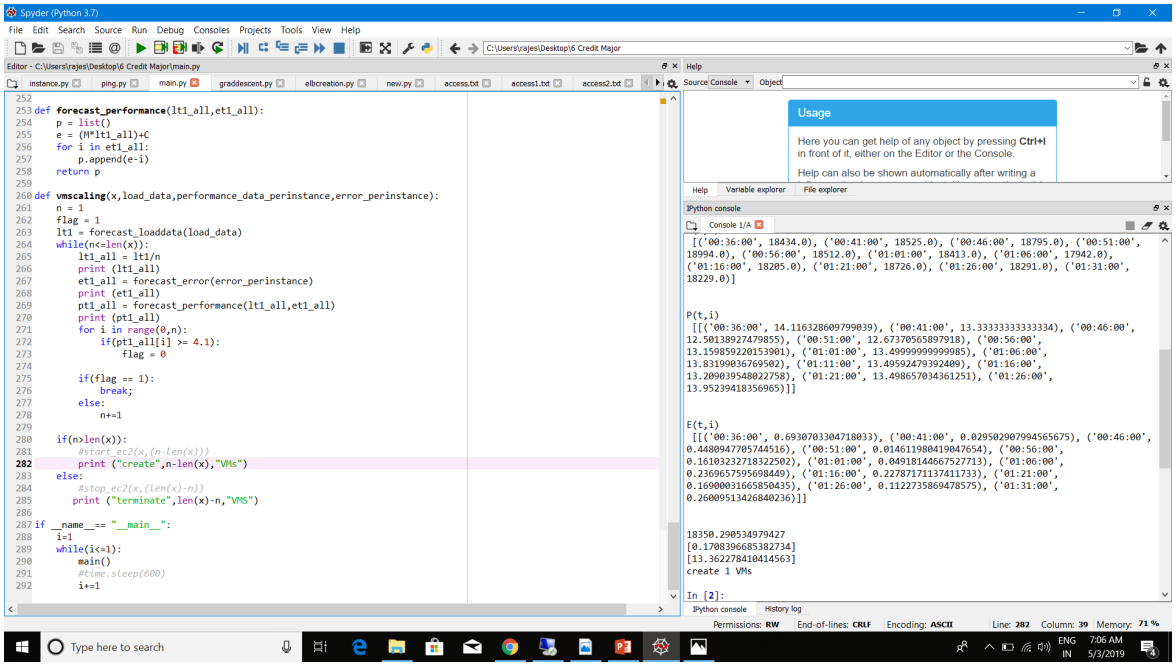

Figure 6.3: Fig. Creation 1 extra VM to divide load

6.2 Case 2(Underutilization): Termination of virtual machines if http requests decrease

1. At first a three Vitual Machine is put to test.

2. Around 300 http requests(reduced) per 5 minutes are given from a virtual machine in the same region where the VM on test is available. Region is kept same so as to avoid latency.

3. Accordingly CPU utilization of the test VMs is recorded for atleast 3hrs continuously.

4. If CPU utilization is less(say 6) the VM are simply underutilized. So, we calculate the number of virtual machine actually required to process the requests and stop other VMs.

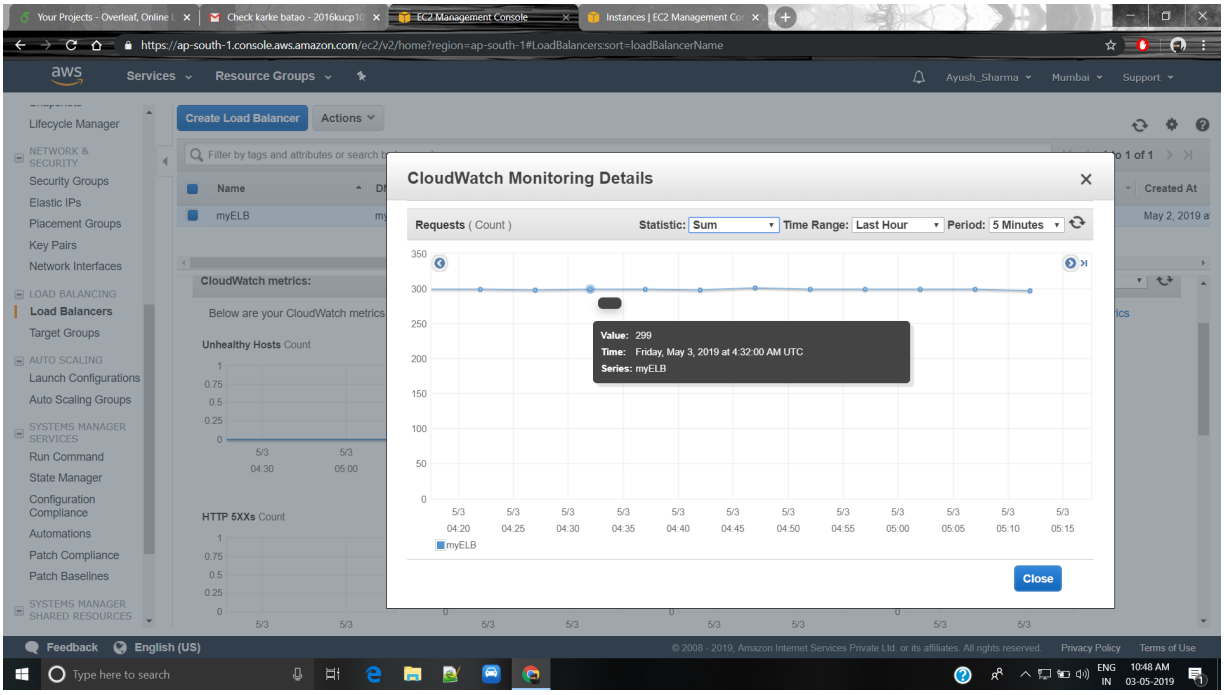

Figure 6.4: Fig. Number of Http Requests

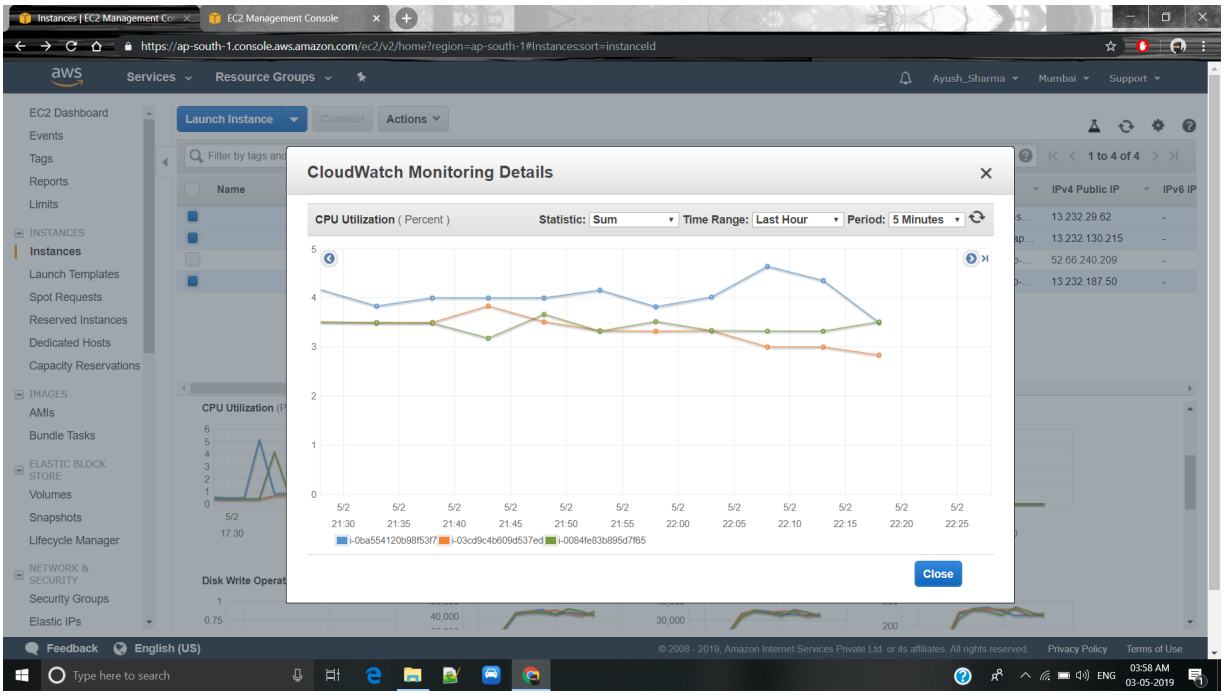

Figure 6.5: Fig. CPU Utilization of Running VMs less than target performance

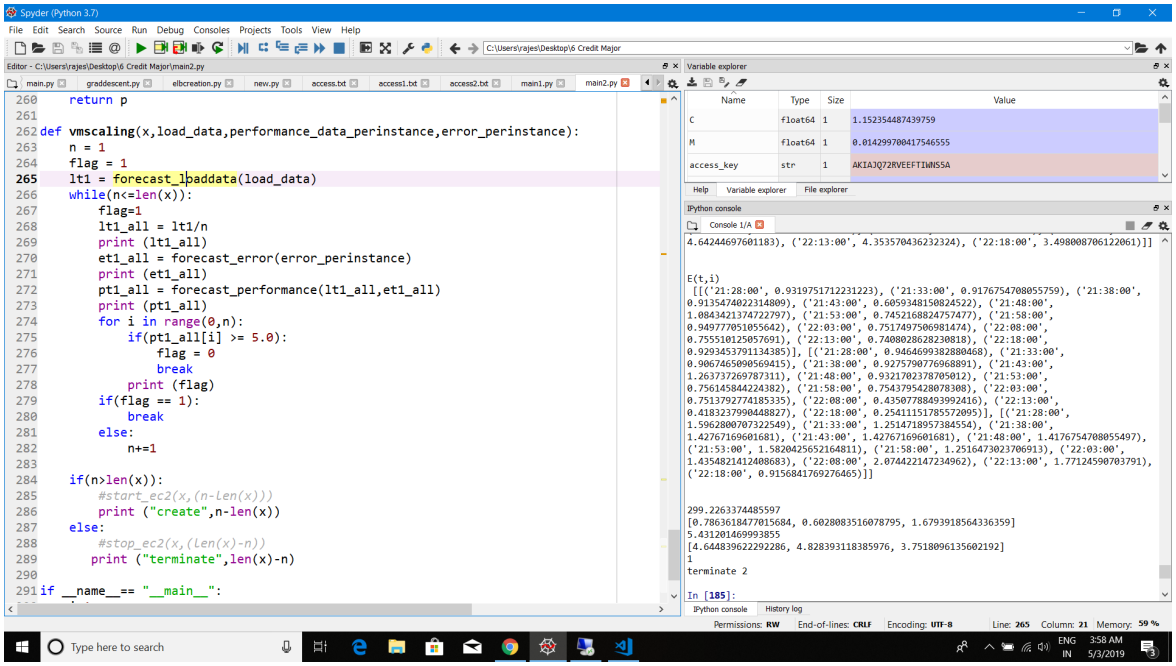

Figure 6.6: Fig. Creating 1 extra VM to divide load

6.3 Findings

If the forecasted performance based on unexpected performance fluctuation:-

1.)exceeds the target performance(any one of the VMs) i.e. CPU Utilization of VM is more leading to less performance then a new VM is created.

2.)is less(of all the VMs) than the target performance then the number of virtual machines required is calculated again and uneccessary VMs are stopped for good.

3.)According to the proposed methods VMs can be created one by one and multiple VMs can terminated at once.

4.)At any time atleast 1 VM will be running.

Chapter 7

Problem Faced During Impementation

7.1 Wait Time Constraints while using AWS Cloud

Since, for forecasting the future values previous data is required a lot of time(2 weeks) was invested collecting the data points. This is due to the 5min refresh rate of Cloudwatch Monitoring Services(Http Requests,CPU Utilization).

7.2 Latency Problems

While Using the facilities of Mumbai region, latency was encountered due to the number of http requests fluctuated vastly thus affecting our assumptions.
To overcome this latency a virtual machine was made in the same region to give http requests also through which we got around 98 percent accuracy.

7.3 Lack of Requests to fully utilize VMs

Working under free tier AWS Cloud Services we were unable to give sufficient requests i.e. 1200000 http requests for 100 percent utilization.
As a result the target performance was kept low deliberately to meet our constraints requirement.

Chapter 8

Conclusion

We proposed a VM scaling method that forecasts the unexpected performance fluctuation to adjust the number of VMs. We evaluated the proposed method and confirmed that the proposed method improved the success rate of message processing.
According to our evaluation VMs are successfully created and terminated according to the required so that neither the resources are over utilized nor underutilized.

Chapter 9

Future Scope

9.1 The project will be extended in future as follows:-

9.1.1 New Methods to calculate performance of VMs.

9.1.2 IoT Connectivity with the VMs to increase the expected load

9.1.3 Improve Load Balancer Allocation Methods

9.1.4 A bill generation scheme based on the use of our resources similar to AWS billing scheme

References

[1] *2017 IEEE 10th International Conference on Cloud Computing*
Virtual Machine Scaling Method Considering Performance Fluctuation of Public Cloud; Yu Kaneko, Toshio Ito, Masashi Ito, Hiroshi Kawazoe

[2] O. Givehchi, H. Trsek, and J. Jasperneite, Cloud computing for industrial automation systems - A comprehensive overview, in Emerging Technologies Factory Automation (ETFA), 2013 IEEE 18th Conference on, Sept 2013, pp. 14.

[3] N. Vasic, D. Novakovic, S. Miucin, D. Kostic, and R. Bianchini, DejaVu: Accelerating Resource Allocation in Virtualized Environments, SIGARCH Comput. Archit. News, vol. 40, no. 1, pp. 423436, Mar. 2012.

[4] H. Nguyen, Z. Shen, X. Gu, S. Subbiah, and J. Wilkes, Agile: Elastic distributed resource scaling for infrastructure-as-a-service, in Proceedings of the 10th International Conference on Autonomic Computing (ICAC 13). USENIX, 2013, pp. 6982.

[5] Y.-H. L. Jin-Cherng Lin and C.-H. Liu, Building time series forecasting model by independent component analysis mechanism, in Proceedings of The World Congress on Engineering 2007, July 2007, pp. 10101015.

[6] Amazon Web Services. Elastic Compute Cloud (EC2) Cloud Server Hosting. [Online]. Available: https://aws.amazon.com/ec2/

[7] M. Mao and M. Humphrey, A performance study on the vm startup time in the cloud, in 2012 IEEE Fifth International Conference on Cloud Computing, June 2012, pp. 423430.

[8] K. Aida, O. Abdul-Rahman, E. Sakane, and K. Motoyama, Evaluation on the performance fluctuation of hadoop jobs in the cloud, in 2013 IEEE 16th International Conference on Computational Science and Engineering, Dec 2013, pp. 159166.

[9] G. Somani and S. Chaudhary, Application Performance Isolation in Virtualization, in Cloud Computing, 2009. CLOUD 09. IEEE International Conference on, Sept 2009, pp. 4148.

[10] X. Pu, L. Liu, Y. Mei, S. Sivathanu, Y. Koh, C. Pu, and Y. Cao, Who Is Your Neighbor: Net I/O Performance Interference in Virtualized Clouds, Services Computing, IEEE Transactions on, vol. 6, no. 3, pp. 314329, July 2013.

[11] H. Kang, J. in Koh, Y. Kim, and J. Hahm, A sla driven vm autoscaling method in hybrid cloud environment, in 2013 15th Asia-Pacific Network Operations and Management Symposium (APNOMS), Sept 2013, pp. 16.

[12] H. Fernandez, G. Pierre, and T. Kielmann, Autoscaling web applications in heterogeneous cloud infrastructures, in 2014 IEEE International Conference on Cloud Engineering, March 2014, pp. 195204.

[13] Y. t. Lee, W. h. Hsiao, C. m. Huang, and S. c. T. Chou, An integrated cloud-based smart home management system with community hierarchy, IEEE Transactions on Consumer Electronics, vol. 62, no. 1, pp. 19, February 2016.

[14] Erlio GmbH. VerneMQ. [Online]. Available: https://vernemq.com