Computers, Materials & Continua
DOI:10.32604/cmc.2022.019344
Article

Tech Science Press

Load Balancing Framework for Cross-Region Tasks in Cloud Computing

Jaleel Nazir[1,2], Muhammad Waseem Iqbal[1], Tahir Alyas[2], Muhammad Hamid[3], Muhammad Saleem[4], Saadia Malik[5] and Nadia Tabassum[6,*]

[1]Department of Software Engineering, The Superior College, Lahore, Pakistan
[2]Department of Computer Science, Lahore Garrison University, Lahore, 54000, Pakistan
[3]Department of Statistics and Computer Science, University of Veterinary and Animal Sciences, Lahore, 54000, Pakistan
[4]Department of Industrial Engineering, Faculty of Engineering, Rabigh, King Abdulaziz University, Jeddah, 21589, Saudi Arabia
[5]Department of Information Systems, Faculty of Computing and Information Technology - Rabigh, King Abdulaziz University, Jeddah, 21589, Saudi Arabia
[6]Department of Computer Science, Virtual University of Pakistan, Lahore, 54000, Pakistan
[*]Corresponding Author: Nadia Tabassum. Email: nadiatabassum@vu.edu.pk
Received: 10 April 2021; Accepted: 24 May 2021

Abstract: Load balancing is a technique for identifying overloaded and underloaded nodes and balancing the load between them. To maximize various performance parameters in cloud computing, researchers suggested various load balancing approaches. To store and access data and services provided by the different service providers through the network over different regions, cloud computing is one of the latest technology systems for both end-users and service providers. The volume of data is increasing due to the pandemic and a significant increase in usage of the internet has also been experienced. Users of the cloud are looking for services that are intelligent, and, can balance the traffic load by service providers, resulting in seamless and uninterrupted services. Different types of algorithms and techniques are available that can manage the load balancing in the cloud services. In this paper, a newly proposed method for load balancing in cloud computing at the database level is introduced. The database cloud services are frequently employed by companies of all sizes, for application development and business process. Load balancing for distributed applications can be used to maintain an efficient task scheduling process that also meets the user requirements and improves resource utilization. Load balancing is the process of distributing the load on various nodes to ensure that no single node is overloaded. To avoid the nodes from being overloaded, the load balancer divides an equal amount of computing time to all nodes. The results of two different scenarios showed the cross-region traffic management and significant growth in revenue of restaurants by using load balancer decisions on application traffic gateways.

Keywords: Load balancing; performance; database; region; virtualization

1 Introduction

The purpose of a delivery platform is to deliver applications, computing power, and storage. In cloud computing users can access and retrieve data from any platform at all times, through the internet. Using Load balancing helps to distribute all loads (traffic) that come from the client-side divided into nodes. It also ensures that each computing resource is distributed efficiently and fairly for users, so that they do not face any problems accessing their data. In the past few years, cloud computing has provided these facilities as a utility to meet the everyday needs of the general public. Nowadays cloud computing has become the most popular and fast-growing technology in the IT industry [1]. Infrastructure as a Service (IaaS), platform as a Service (PaaS), and Software as a Service (SaaS) are three prominent types of services offered by the cloud providers. Cloud computing also provides several features to assist you to unlock new workloads like Content-based routing, Container support, and Application monitoring [2]. In Fig. 1 showed different load balancing groups like blog, web servers and forums group connected through database and controlling the traffic with load balancer.

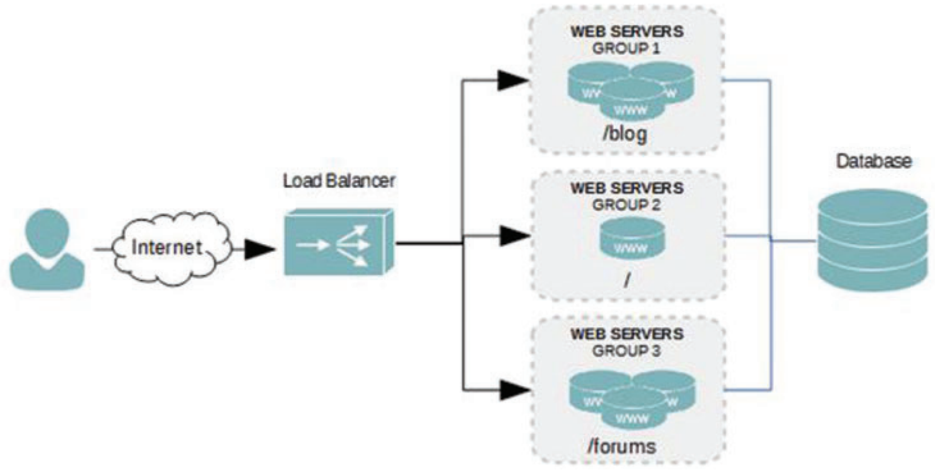

Figure 1: Load balancing groups

2 Related Work

In focus on reducing service response, time, and performance of load balancing Techniques for Efficient Traffic Management cloud computing. In this work get a result to spot the challenges in Load balancing for Autoscaling in AWS with a different kind of application workload. In optimizing load balancing algorithm [3] Meta Heuristics approach is employed to boost the performance of Service Level Agreement (SLA) may be a service that helps communication between a service provider and a customer [4]. In providing a help study for a way you'll manage existing load balancing algorithms in cloud computing. Using AWS offerings, specifically for EC2, S3, and Amazon's CDN Cloud Front the results of big workload unbalance among different datacenters hosting both EC2 and S3 products; specifically, the info center in Virginia is answerable for 85% traffic sent to Italian end-users and it handles seven times traffic served by the info center in Ireland [5]. In Result investigates that there is a conflicting increment in-network presence of normal gadget even as improved data exactness and small measurements are ready and put away in cloud servers with high precision, subsequently upgrading framework balance by the use of burden adjusting strategy. In conclusion from the simulation results that the total cost is incredibly well optimized by the utilization of closest data Centre policy or optimized latency policy as compared to reconfigure dynamically along with throttled load balancing [6].

In today's large-scale computing systems, heterogeneity is becoming more common. It's critical to develop effective load balancing policies for systems with differing resource speeds if you want to achieve fast response times. Indeed, how to better distribute jobs to servers is a well-known and well-studied topic in queueing theory. However, the vast majority of current research on large-scale systems is based on homogeneous servers, policies that perform well in a homogeneous environment will result in unacceptably low results in heterogeneous systems. Dense peak traffic periods and wild swings in traffic patterns end in low utilization rates of high-priced hardware, yielding high operating costs to maintain idle hardware, similarly as an inefficient use of capital for underutilized hardware [7].

Serverless computing is one of the latest terms concerned with cloud computing. Load balancing played an important role in Cloud computing applications. There are different algorithms for improving the load balancing technique like Static Load balancing algorithm (SLBA), Dynamic load balancing algorithms (DLBA), and dynamic nature-inspired load balancing algorithm (NDLBA). The results proved that NDLBA and DLBA are efficient in their working [8].

With the increase in cloud data, issues regarding load balancing are appearing. These issues can be resolved by the use of algorithms. These algorithms include task scheduling, resource management, quality of service, and workload management. These are some of the suggested methods which can be used with the load balancing algorithm to make it more effective [9]. In E-commerce, the huge company's business process is sub-divided into sub-business processes. This business process is called a distributed workflow management system. There are several load balancing algorithms to manage load in the DWMS environment i.e., round robins and load aware scheduling algorithm [10]. The results suggested that round robins are best in uniform workflow systems and load ware in a distributed system. Cloud computing systems should be scalable because cloud computing is a need of every business nowadays. The major purpose of this system is to create ease for users. The load balancing algorithms can be checked by the use of several characteristics like resource utilization, response time, performance, overhead, fault tolerance [11]. Load balancing is not only concerned with the management of user requests for data. But load balancing is also concerned with the memory load, CPU load, and network load etc. There are different types of cloud services that are available. The user depends on the characteristics of the services, which must be taken into account while doing a selection of services.

Data access and data storage in cloud computing is done through the network. The user doesn't have any access to the security management system of the cloud-based services. Several security challenges must be catered to by implementing the true security management system. The Internet of Things (IoT) is also a term similar to cloud computing. In the IoT, things will be made smart by the use of the internet, but, the management of the cloud services is one of the major challenges here. The performance check can be made by QoS and some factors like power utilization, Makespace, and execution time. During the data access and data storage on the cloud, the requests are numerous in nature. Maybe, some requests required a shorter period, and some requests required a longer time period. So, this delay in processing can affect load balancing techniques in cloud computing. Load balancing is a favorite term for every business. Because no management of resources and services is required. It is the headache of the service provider. But still, there are a number of issues regarding cloud services like data security, storage charges, capacity, load balancing and service brokering. The water flow model is also one of the main load balancing algorithms, the performance of this algorithm is far better than the Genetic algorithm [12].

Efficient use of resources and to obtain energy efficiency is also one of the issues concerned with cloud computing. Ant colony system algorithm used to manage resources more effectively, and to manage underutilized server machines. With the availability of easy cloud services, the use and thus the load on the cloud infrastructure is increasing day by day. The cloud services due to fewer expenses has more attraction but still, the cloud services are not as efficient. The response time and threshold are the major problems while handling load balancing. There are a number of algorithms available for this purpose. The Shortest job first scheduling can be used to achieve better response time [13].

Load balancing, in Cloud Computing (CC) environment, is defined as the method of splitting workloads and computing properties. It enables enterprises to manage workload demands or application demands by distributing the resources among computers, networks, or servers.

There are a number of algorithms available to solve the load balancing problem, for example, Load balancing improves the Min-Min algorithm (LBIMIM), improved backfill algorithm. These algorithms may raise problems of response time. This problem can be solved by the use of a balanced spiral with already available algorithms. Particle Swarm Optimization (PSO) is also one of the algorithms which can be integrated with the load balancing algorithms to increase load balancing and to schedule jobs in the queue. Cloud computing services are accessible through the link. The security issues and the proper energy utilization in cloud computing machines are increasing and thus results in greenhouse gases. The best-fit decreasing was proposed to counter these two issues in cloud computing. This algorithm will help to improve energy efficiency and to improve Service Level Agreement [14].

3 Proposed Methodology

The proposed load balancing model consists of an automatic scaling listener, application module, migration module monitoring and CPU performance module. Mostly use of EC2 or Elastic beanstalk instances for any HTTP and HTTPS based applications that can create for one instance so the upcoming traffic from client-side can hit only one load balancer and also we setting the properties of the load balancer as shown in Fig. 2. Automatic scaling lister is responsible for dynamic scaling, the automatic scaling listener works as a system of data agent that controls and records traffic between cloud service users and cloud providers. The automatic scaling listener expands the service and starts the development of redundant instances.

The automatic scaling listener is divided into three sub-modules monitor, analyzer, and planner under the knowledge base. The automatic scaling listener module is connected with the Application module and monitoring module. All network traffic and physical machine that monitors the CPU performance during the execution of application module which contains the web application, database and monitoring module of different region server. The application module is connected with the migration module for load balancing decisions. By keeping all the knowledge base of decision and load of traffic on cloud application migration module will take the decision and shift the traffic. The migration module is divided into three sub-modules for decision-making.

- VM Monitoring
- Under/over-utilize VM

- Load Balancing Decision
- Migration Decision

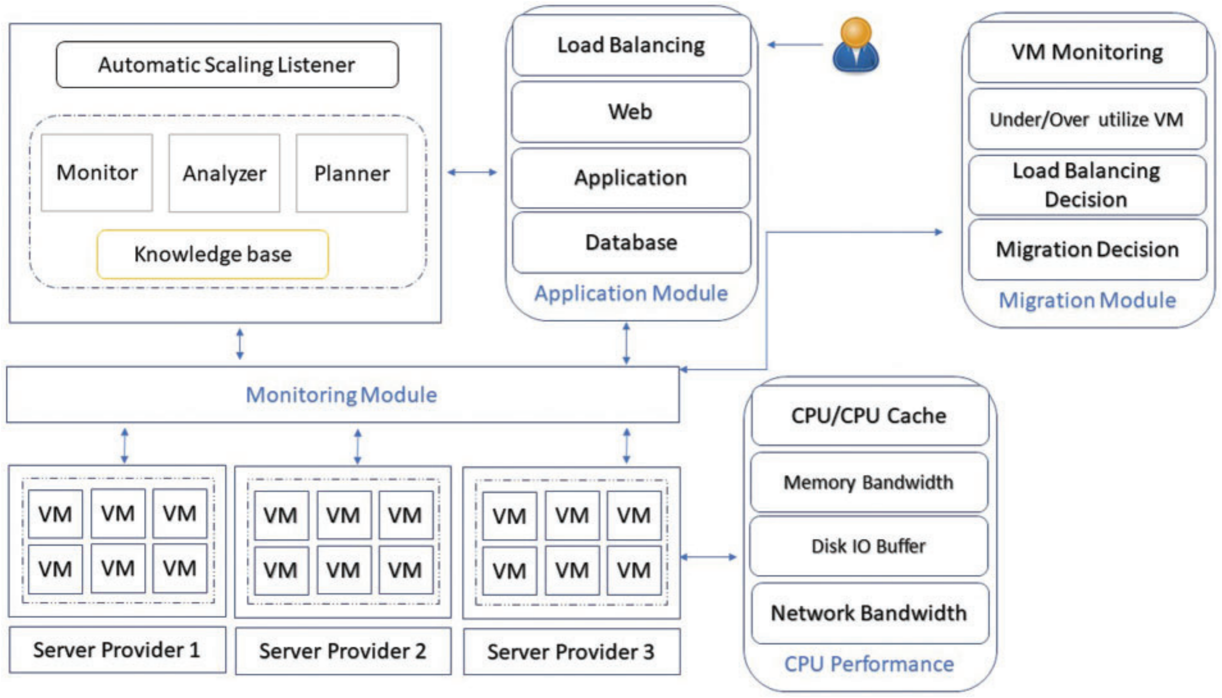

Figure 2: Proposed load balancing model

All VM monitoring is done through an automatic scaling listener if network traffic is under over-utilize then migration decision on the basis of load balancing decision to shift the traffic to next available resources from the providers.

CPU performance module is further divided into four sub-modules

- CPU/CPU Cache
- Memory Bandwidth
- Disk IO Buffer
- Network Bandwidth

CPU performance is responsible for the monitoring of CPU cache, memory bandwidth, IO buffering, and network performance. All parameters will report the hardware layer state to the monitoring module for decision-making.

Two different problems for the cross-region task has been discussed to validate the problem.

- Restaurant Load balancing problem
- Multiple database API for cross-region problem

In the first scenario the main objective of this paper to know if I want to maintain upcoming traffic for every restaurant of my product and specify which restaurant is most valuable or beneficial for more revenue generation. To cater the ordering problem in restaurant produces more income or most popular and how many people approach which restaurant for orders. In this scenario proposed system will check every restaurant traffic coming from the app side using two methods make domains for every restaurant or setting proxy with target groups to make setting URL for every restaurant and when a specific target URL hit for a specific restaurant this hit URL maintain own traffic using separate elastic load balancing [15]. Most applications face the issue of handling load when users grow exponentially, initially when the application was developed, the development team did not notice these kinds of issues for medium-scale systems, but eventually, this issue occurs at some point. But at this point when the application is live and working in the market then, this is not a good choice to develop the whole system again, so the team has to use some technique by which the application starts working smoothly and in the parallel team can develop new system [16].

In the second scenario the cross-region solution of making the existing system smooth, make the DB's separate for different regions where the application is live and the system will detect the using IP then decide from which DB to connect. when a user request some resource on the application, the request will first go to the API hub then the API hub will decide which DB to connect by checking the request IP, IP contains the information about the user's region. With the passage of time and due easiness of cloud services, businesses are shifting towards internet-based services. The business through the internet is growing and e-commerce based applications using this strategy to enhance their productivity. There is a large number of companies in the large volume that are running their business through the internet.

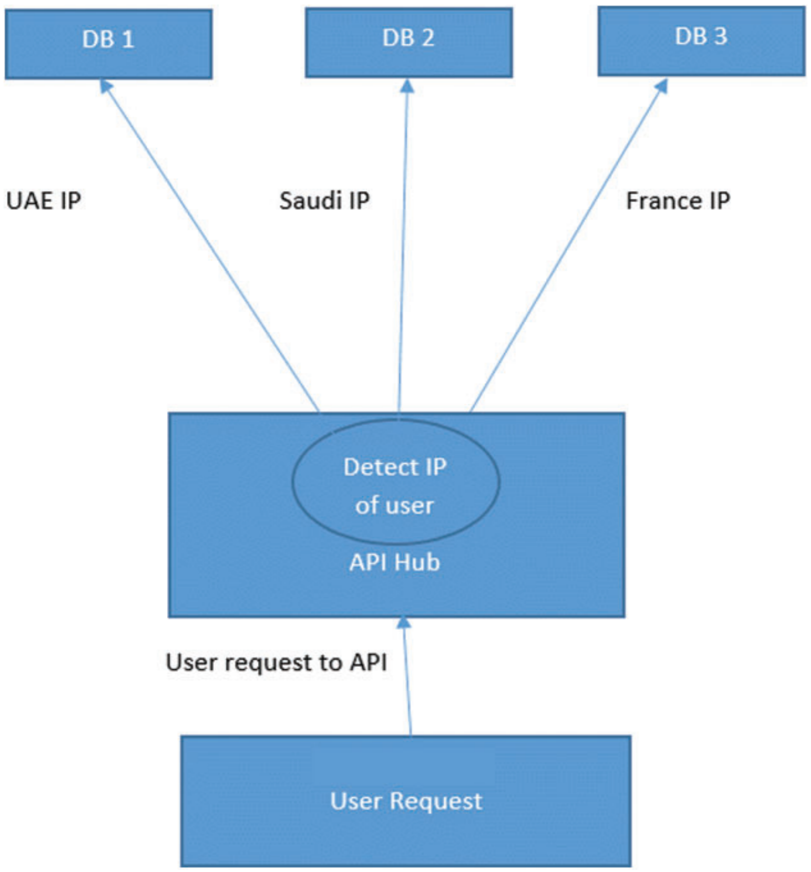

Figure 3: Multiple databases API

The companies have their business running in the whole world. i.e., Alibaba which is one of the largest online shopping websites in the world. The users accessing this website are in billions. The users may belong to a different part of the world. If a company running an online business has its main system in one region which means companies cloud-based data storage is in one region. But, the billions of users are accessing their service belong to different regions of the world. Then, if all the requests received in the system is processed by the main system then the system will be overloaded and the system may crash. This results in a huge loss for the users and also for the providers. The second scenario research problem is based on the study of the e-commerce issues regarding load balancing, how e-commerce is working, and what is the solution for the E-commerce load balancing is working for multiple databases API is shown in Fig. 3. User request through API hub and then detect IP and redirect the traffic to the concerned region and connect that region database.

4 Proposed Algorithm

Working algorithm of load balancing over db connections

```
Operational_regions_Array = ['UAE', 'FRANCE', 'SAUDI ARABIA']
Request_IP = get request IP with built in method
Request_region = get request region from Request_IP using built in method
If (Operational_regions_array contain Request_region)
// connect request with Request_region db
Else
//Request denied
Endif
```

When developing a new application, if the team considers the user exponential growth in the future, then the team has to build the application's architecture flexibly so that there will be no effect of user growth on the system.

The one way of doing this is modeled in Fig. 4. According to this model, there will be separate request handlers for different regions. Mean there will be a separate domain for separate regions. Suppose the user is requesting from the UAE region then the user will connect to the UAE domain and this domain has a separate copy of the codebase which is responsible to give a response accordingly. Also, the DB is separate for different regions, so one domain only bears a load of the domain's specific region in this way the application can manage load efficiently and effectively.

Working algorithm of Load balancing over regions

```
Operational_regions_Array = ['UAE', 'FRANCE', 'SAUDI ARABIA']
Request_IP = get request IP with built in method
Request_region = get request region from Request_IP using built in method
If (Operational_regions_array contain Request_region)
// connect request with Request_region API_HUB
// API_hub have own connected db
Else
```

```
//Request denied
End
```

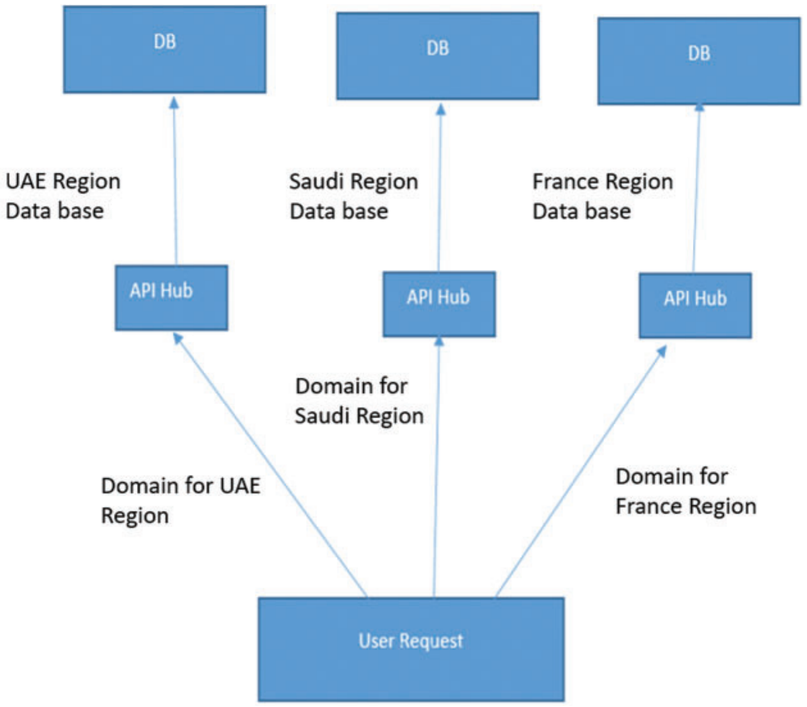

Figure 4: Separate domain for separate regions

5 Results & Discussion

For 1st scenario of the restaurant problem, https://www.restaurant.com architecture is used for decision making when a client has decided to segment services such as processing orders for restaurants like kfc macdonald and bundu khan in the Pakistan region. Each function expressed a discrete collection of instances and also each collection of instances host several applications that provide a cloud service.

Using a classic load balancer, the customer needs to deploy several load balancers. Every load balancer points to the instances that represent and front the service by using a sub-domain. Application Load Balancers explain the concept of rules, targets, and target groups. Rules determine setting the target for specific URL hit from browser or app side. Each rule represents a target group, condition, and priority and action is taken when the conditions on a rule are matched for example if the order is placed from kfc restaurant then target hit this https://www.restaurant.com/kfc this means that condition is matched and call this domain. Targets are endpoints that may be registered as a member of a target group. Target groups are wont to route requests to registered targets as a part of the action for a rule as shown in Fig. 5.

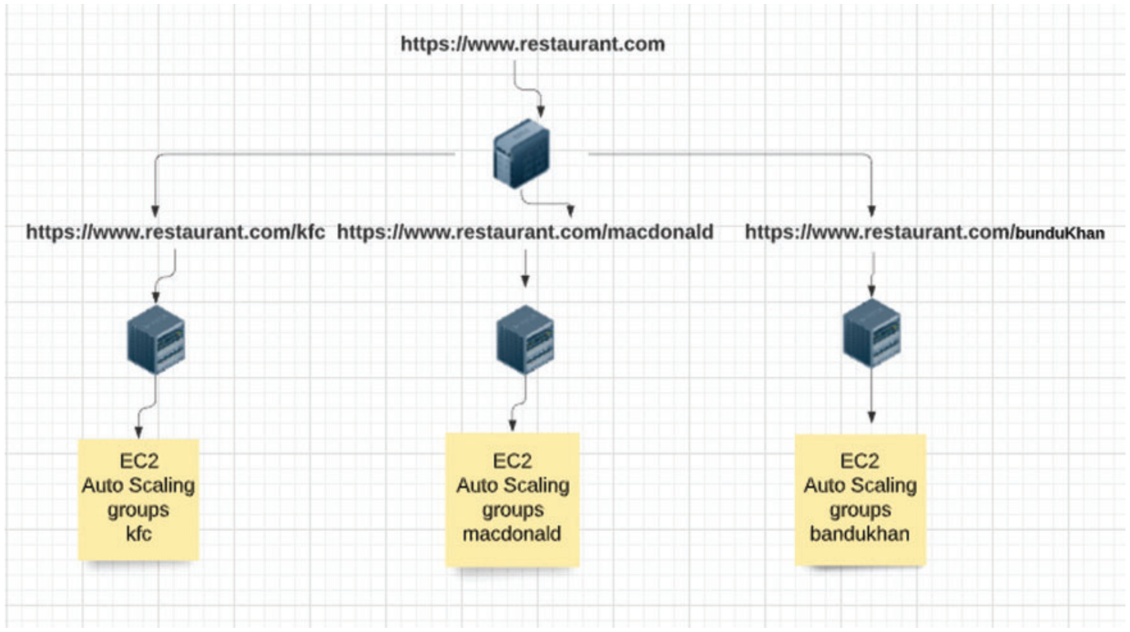
https://www.restaurant.com
https://www.restaurant.com/kfc
https://www.restaurant.com/macdonald
https://www.restaurant.com/bunduKhan
EC2 Auto Scaling groups kfc
EC2 Auto Scaling groups macdonald
EC2 Auto Scaling groups bandukhan

Figure 5: Service segmentation using subdomains

In this implementation, https://www.restaurant.com is connected with a top-level external load balancer. This load balancer is configured with 3 subdomains (/restaurant name (KFC)) so that traffic is distributed to a group of instances running. Each instance running NGINX is configured with rules that direct traffic to one of the three internal load balancers based on the path in the URL. For example, when a customer can place an order from app side to KFC restaurant then hit and matches a location for https://www.restaurant.com/kfc and sends traffic to the server and inside the load balancer fronting the group of servers providing the Amazing restaurant functionality In this implementation, proposed system is able to create three rules that time to different target groups supported different path conditions as shown in Fig. 6.

For revenue calculation, proposed system calculated restaurant revenues and most number of order received during the year 2020 to check which restaurant gets more traffic using calculate more order placing of every restaurant by displaying the graph in our product. The graph showing the web panel of Sichuan hotel received 140 orders at most and Rizwan Burger restaurant receive at lower order during year 2020 as shown in Fig. 7.

For the revenue generation term, the most revenue generated by Sichuan by 8000 thousand and KFC receive revenue of 2000 thousand in as shown in Fig. 8.

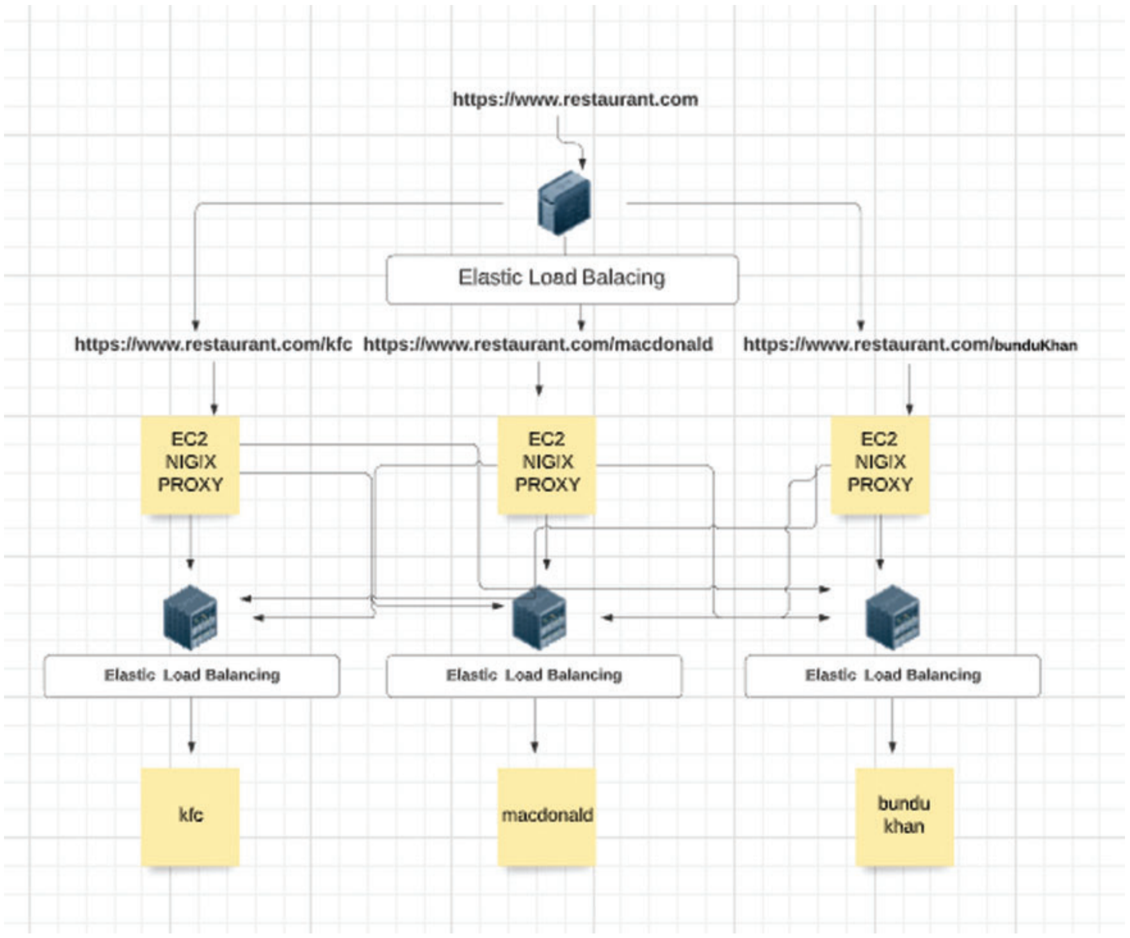

Figure 6: Service segmentation using proxy layer

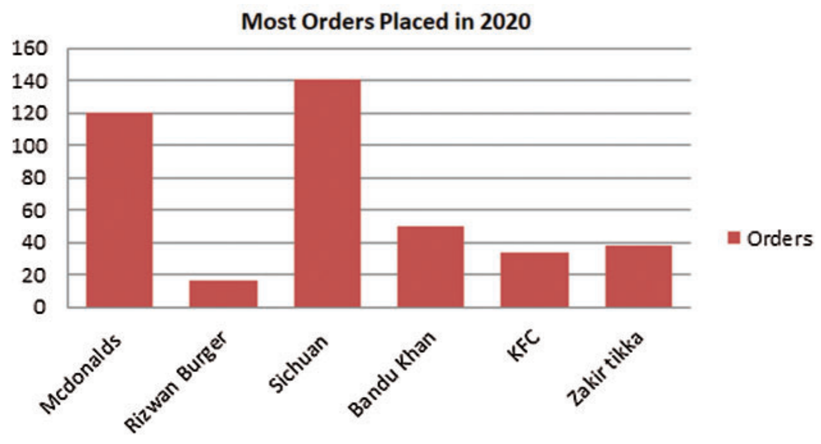

Figure 7: Most order place

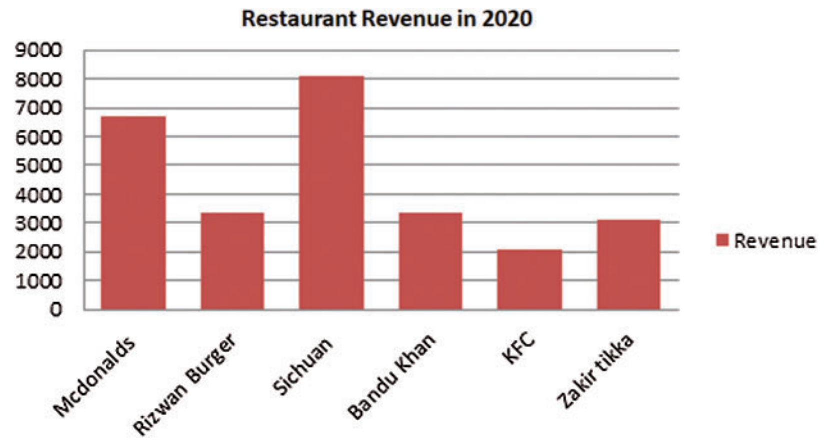

Figure 8: Revenue generated

For the second scenario of the multi-database for cross-region in term of computation time, the system tested API to fetch only first page products for both architectures the results made a very clear difference that, if some application uses a different domain for the different region then response time is very low mean this architecture is very fast. The results when product fetch API is hit in "same domain different DB" architecture it consume 8 s to respond. The results, when same products fetch API hit in "different domain for different regions" architecture, it consumes 2 s to respond. So the difference is clear and very big in the response time of both architectures.

6 Conclusion

Load balancing is used for efficient usage of cloud resources and helpful for sharing computing workload across multiple regions. Automatic scaling listener monitored the network traffic and spread the dynamic load equally across multiple cross-region for better computing load. Exploring efficient proposed load balancing algorithms that can detect IP through API hub to achieve improved access time in different regions. In restaurants revenue method the use of Elastic Load Balancing reduces the operational overhead and monitors the network traffic for the different domain-oriented restaurants for most order taking and revenue sharing purposes. Results showed that the use of load balancing over database connection is more cost-efficient and generated more revenue for the year 2020.

Acknowledgement: Thanks to our teachers and families who provided moral support.

Funding Statement: The authors received no specific funding for this study.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] H. Yang, Q. Zhao, Z. Luan and D. Qian, "IMeter: An integrated VM power model based on performance profiling," *Future Generation Computer System*, vol. 36, no. 6, pp. 267–286, 2014.
[2] A. Evangelidis, D. Parker and R. Bahsoon, "Performance modeling and verification of cloud-based auto-scaling policies," *Future Generation Computer System*, vol. 87, no. 4, pp. 629–638, 2018.

[3] B. Mao, Y. Yang, S. Wu, H. Jiang and K. Li, “IOFollow: Improving the performance of VM live storage migration with IO following in the cloud,” *Future Generation Computer System*, vol. 91, no. 2, pp. 167–176, 2019.

[4] V. Simic, B. Stojanovic and M. Ivanovic, “Optimizing the performance of optimization in the cloud environment: An intelligent auto-scaling approach,” *Future Generation Computer System*, vol. 101, no. 23, pp. 909–920, 2019.

[5] B. Bibal and D. Dharma, “HAS: Hybrid auto-scaler for resource scaling in cloud environment,” *Journal of Parallel Distributing Computer*, vol. 120, no. 7, pp. 1–15, 2018.

[6] S. Saharan, G. Somani, G. Gupta, R. Verma, M. Gaur *et al.*, “Quickdedup: Efficient VM deduplication in cloud computing environments,” *Journal of Parallel Distributing Computer*, vol. 139, no. 1, pp. 18–31, 2020.

[7] Y. Cheng, W. Chen, Z. Wang, Z. Tang and Y. Xiang, “Smart VM co-scheduling with the precise prediction of performance characteristics,” *Future Generation Computer System*, vol. 105, no. 23, pp. 1016–1027, 2020.

[8] D. Saxena and A. K. Singh, “A proactive autoscaling and energy-efficient VM allocation framework using online multi-resource neural network for cloud data center,” *Neurocomputing*, vol. 426, no. 3, pp. 248–264, 2021.

[9] M. Hamid, N. Björsell and S. Ben Slimane, “Signal bandwidth impact on maximum-minimum eigenvalue detection,” *IEEE Communication Letters*, vol. 19, no. 3, pp. 395–398, 2015.

[10] K. Ye, H. Shen, Y. Wang and C. Xu, “Multi-tier workload consolidations in the cloud: Profiling, modeling and optimization,” *IEEE Tranaction on Cloud Computing*, vol. 71, no. 3, pp. 1–9, 2020.

[11] M. Shifrin, R. Mitrany, E. Biton and O. Gurewitz, “VM scaling and load balancing via cost optimal MDP solution,” *IEEE Tranaction on Cloud Computing*, vol. 71, no. 3, pp. 41–44, 2020.

[12] M. Ciavotta, G. Gibilisco, D. Ardagna, E. Nitto, M. Lattuada *et al.*, “Architectural design of cloud applications: A performance-aware cost minimization approach,” *IEEE Tranaction on Cloud Computing*, vol. 71, no. 3, pp. 110–116, 2020.

[13] P. Kryszkiewicz, A. Kliks and H. Bogucka, “Small-scale spectrum aggregation and sharing,” *IEEE Journal Selected Areas Communication*, vol. 34, no. 10, pp. 2630–2641, 2016.

[14] G. Levitin, L. Xing and Y. Xiang, “Reliability vs. vulnerability of N-version programming cloud service component with dynamic decision time under co-resident attacks,” *IEEE Transaction Server Computing*, vol. 1374, no. 3, pp. 1–10, 2020.

[15] M. Aslanpour, M. Ghobaei and A. Nadjaran, “Auto-scaling web applications in clouds: A cost-aware approach,” *Journal Network Computer Application*, vol. 95, pp. 26–41, 2017.

[16] V. Roussev, I. Ahmed, A. Barreto, S. McCulley and V. Shanmughan, “Cloud forensics–tool development studies & future outlook,” *Digital Investigation*, vol. 18, no. 3, pp. 79–95, 2016.