Implementation of Application Load Balancing and Auto Scaling using AWS

[1]V.Geetha Lakshmi[2]Mr.E.Ramesh Reddy

[1]CSE, NEC, Gudur

2Associate Professor,CSE Department, NEC, Gudur

Abstract: *Most of the industries maintaining their web server on their on-premise, which leads them to have high maintenance, more workload, increases cost. To overcome this we can use cloud technologies for building a web server which gives better performance. Thus we use AWS cloud technologies. A web server is built using the cloud formation technique. To achieve these major cloud resources are used such as Ec2 Instance, Application load balancer for balancing the load given by users, auto-scaling group. Elastic Load Balancing automatically divides incoming application traffic over different targets, like Amazon EC2 instances, containers, IP addresses. It can manage the differing load of your application traffic in one Availability Zone or over different Availability Zones. Elastic Load Balancing offers three types of load balancers which all presents the high availability, automatic scaling, and robust security required to make your applications. AWS Auto Scaling monitors your applications and automatically adjusts capacity to maintain steady, predictable performance at the lowest possible cost. Using AWS Auto Scaling, it's easy to setup application scaling for multiple resources across multiple services in minutes.*

Keywords: *AWS, EC2 Instance, Load balancer, Auto Scaling group*

I. INTRODUCTION

With more and more advances being made in cloud computing and its increasing efficiency, companies have started using cloud as their underlying architecture for most of the important operations. The demand for resources is always increasing in these companies and with the help of cloud architecture, all the demand requirements are met easily. Cloud allows them to increase/decrease the load on servers according to their requirements as cloud provides the policy of pay-as-you-go which makes it a good option for the organizations. Cloud computing is the on demand delivery of IT resources via the internet, with pay-as-you-go pricing. Instead of buying, owning and maintaining physical data centers and servers, the companies can leverage this task to third party companies to do the same for them. These third-party companies then provide a virtual environment to the user/company which then the user can use without bothering about the details of the architecture/mechanism. Following are a few reasons why every company should migrate to cloud computing.

Load balancing plays a very important role in the networking technology, Load balancer comes into play when the user tries to connect to the server. Any requests made to the internet will reach the server in various paths. There may be N number of servers which are serving the purpose of the request but the request will go to only one depending upon the traffic. The traffic here refers to how busy server is, in responding the requests made by servers. Requests will be directed towards servers by one of the networks called Load balancer which balances the load of server. There are different types of load balancer which handles the traffic in different ways. Major types of Load balancer are Static and Dynamic Load balancer. Amazon EC2 instances, containers, and IP addresses can be automatically distributed by ELB to handle incoming application traffic. ELB can handle a single Availability Zone or a number of Availability Zones. ELB has features such as automatic scaling, robust security, and high availability. It is possible to use a Load Balancer to perform basic load balancing on both layers 4 and 7 at the same time.

This work is licensed under a Creative Commons Attribution 4.0 International License

II. LOAD BALANCING TOOLS

- Cloud Analyst
- Cloud Project
- Amazon Web Service **Cloud Analyst:**

Cloud Analyst is a specialized role within organizations responsible for analyzing and optimizing cloud resources, costs, and performance. Cloud Analysts leverage various tools and methodologies to monitor, analyze, and optimize cloud infrastructure, ensuring efficient resource allocation and cost management. Key Responsibilities of Cloud Analyst:

Cloud Resource Optimization: Cloud Analysts analyze cloud resource utilization and performance metrics to identify inefficiencies and optimize resource allocation for improved performance and cost-effectiveness.

Cost Management: Cloud Analysts monitor cloud spending, identify cost-saving opportunities, and implement cost optimization strategies to reduce overall cloud expenses.

Amazon Web Services (AWS):

Amazon Web Services (AWS) is a leading cloud computing platform that offers a broad set of services, including compute, storage, networking, databases, analytics, machine learning, and more. AWS provides organizations with scalable, flexible, and secure cloud infrastructure to build and deploy a wide range of applications and services.

Key Services and Features of AWS:

Compute Services: AWS offers a variety of compute services, including Amazon EC2 (Elastic Compute Cloud) for virtual servers, AWS Lambda for serverless computing, and Amazon ECS (Elastic Container Service) for containerized applications.

Storage Services: AWS provides scalable storage solutions such as Amazon S3 (Simple Storage Service) for object storage, Amazon EBS (Elastic Block Store) for block storage, and Amazon Glacier for archival storage.

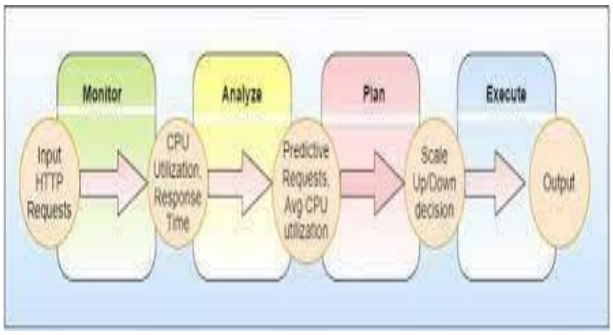
III. IMPLEMENTATION

Following are the steps followed to implement load balancing and auto scaling.

1. Configure a Load Balancer and a Listener
2. Configure Security Settings for an HTTPS Listener
3. Configure a Target Group
4. Create the Load Balancer
5. Create an AMI
6. Configure a Template
7. Create an Auto Scaling Group

MAPE (Monitor, Analyze, Plan, Execute)

This work is licensed under a Creative Commons Attribution 4.0 International License



Launch an EC2 Instance

Amazon Elastic Compute Cloud (EC2) is a web service that provides resizable compute capacity in the cloud. To launch an EC2 instance (i.e., a virtual server), follow these steps:

- Navigate to the EC2 dashboard within the AWS Management Console.
- Click on the "Launch Instance" button to begin the instance creation process.
- Choose an Amazon Machine Image (AMI), which is a pre-configured template for your server. You can select from a variety of operating systems and software configurations.
- Select an instance type based on your resource requirements.
- Configure instance details such as the number of instances, network settings, and storage options.
- Optionally, add tags to your instance for easier management and organization.
- Configure security groups to control inbound and outbound traffic to your instance. Define rules for protocols, ports, and IP ranges.

Access Your Server:

Once your EC2 instance is running, you can access it using SSH (Secure Shell) for Linux instances or RDP (Remote Desktop Protocol) for Windows instances. Use the key pair you created during instance launch to authenticate the SSH or RDP session.

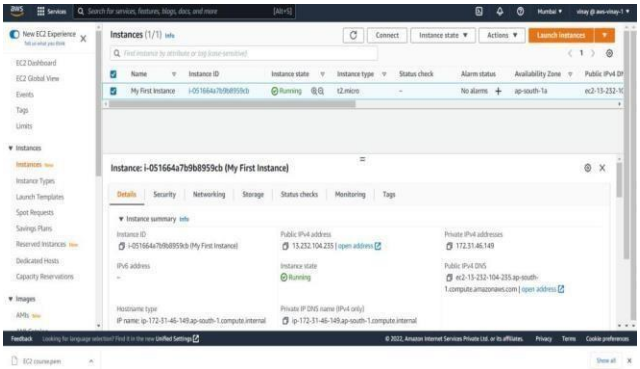


Fig-1 EC2 instance

Ec2 instance with nginx:

Setting up an EC2 instance with Nginx involves a series of steps to launch a virtual server on Amazon Web Services (AWS) and install and configure the Nginx web server software. First, sign up for an AWS account if you haven't already, and access the AWS Management Console. From there, navigate to the EC2 dashboard and click on "Launch Instance". Choose an appropriate Amazon Machine Image (AMI), such as a Linux distribution compatible with Nginx. Select an instance type based on your requirements, configure instance details including network settings and storage options, and set up security groups to allow inbound traffic to the instance. Next,

This work is licensed under a Creative Commons Attribution 4.0 International License

connect to your EC2 instance using SSH and install Nginx. Depending on the Linux distribution, use package managers like yum or apt to install Nginx. Start the Nginx service once installation is complete.

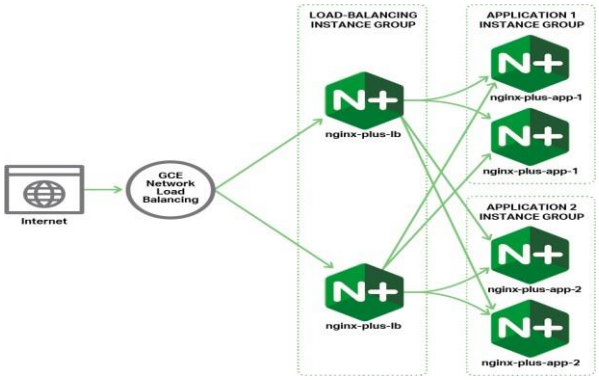


Fig-2 Ec2 instance with nginx

By default, Nginx serves content from the /usr/share/nginx/html directory, which can be replaced with your own website or application. Configure Nginx by creating a new configuration file in the /etc/nginx/conf.d directory and specifying the server block with the appropriate settings, including the server name and root directory. Test the Nginx configuration for syntax errors and reload Nginx to apply the changes. Access your website by entering your EC2 instance's public IP address or domain name in a web browser. Optionally, further configure Nginx for specific requirements such as SSL/TLS certificates, virtual hosts, caching, or security measures. Monitor your EC2 instance and Nginx server regularly for performance, security, and availability, using AWS tools like CloudWatch, and keep your server and Nginx software up to date with the latest patches and updates. By following these steps, you can quickly set up an EC2 instance with Nginx to host your website or web application on AWS.

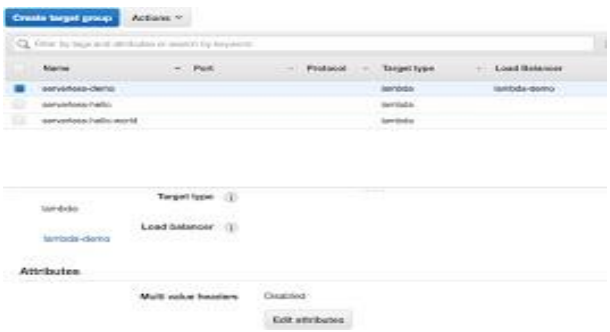


Fig -6 Targets

International Journal of Research in Engineering, IT and Social Sciences, ISSN 2250
6.565, Volume 09 Issue 03, June 2022, Page 4-12

When creating a new target group, the first step is to configure its settings. Start by providing a descriptive name for your target group, ensuring it reflects the purpose or function of the group. Next, select the protocol that the load balancer will use to communicate with the targets. This typically includes options such as HTTP, HTTPS, TCP, or UDP, depending on your specific use case.

This work is licensed under a Creative Commons Attribution 4.0 International License

After defining the basic settings, specify the target type for the group. You can choose between EC2 instances or ECS tasks (containers), depending on the backend resources you intend to route traffic to. If you select EC2 instances, you'll need to choose the Virtual Private Cloud (VPC) where your instances reside.

The next step is to configure health checks for your target group. Health checks are essential for ensuring that traffic is only routed to healthy instances or containers. Define the protocol, port, path, timeout, interval, and threshold for the health checks. This includes specifying the endpoint that the load balancer will periodically check to determine the health of the targets.

Optionally, you can configure advanced health check settings, such as specifying custom success and failure codes or setting a grace period for newly registered targets. These settings provide additional control over the health check process and can be tailored to suit your application's requirements.

Once you've configured the target group settings, the next step is to register targets with the group. Targets can be individual EC2 instances or ECS tasks within your chosen VPC.

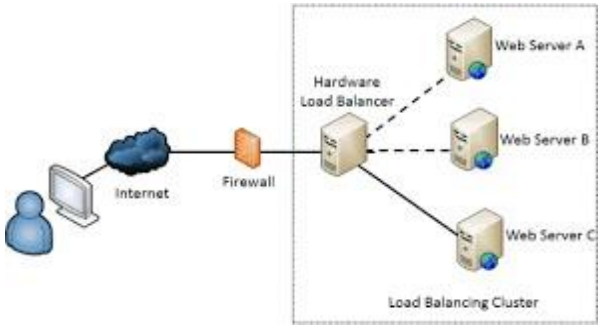


Fig -3 Load balancing

Auto Scaling Group :

An Auto Scaling Group in AWS is a powerful tool for automating the management of EC2 instances and ensuring the availability and scalability of your applications. By following the steps outlined above, you can set up an ASG to dynamically adjust the number of instances in response to changes in demand, improving the reliability and efficiency of your infrastructure.

Scaling policies:

Scaling Policies are attached with the ASG, it tells the ASG when to launch a new server and when to terminate the idle server based on the scaling policies given. Types of scaling policies

1. Target Tracking Scaling

2. Step Scaling

3. Simple Scaling

International Journal of Research in Engineering, IT and Social Sciences, ISSN 2250
6.565, Volume 09 Issue 03, June 2022, Page 4-12

Create AMI:

This work is licensed under a Creative Commons Attribution 4.0 International License

Creating an Amazon Machine Image (AMI) in Amazon Web Services (AWS) allows you to capture the configuration of an EC2 instance, including the operating system, installed software, and any customizations you've made, so that you can launch identical instances in the future. To create an AMI, start by logging in to the AWS Management Console and navigating to the EC2 service. From there, select the EC2 instance that you want to use as the basis for your AMI. Once you've chosen the instance, right-click on it and select "Image and templates" from the dropdown menu, then click "Create image". This will initiate the process of creating a new AMI based on the selected instance.

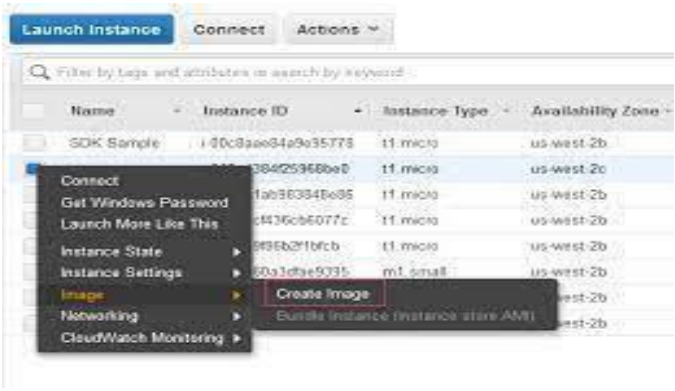


Fig-4 Create AMI

Configure Launch template:

Configuring a Launch Template in Amazon Web Services (AWS) enables you to define the configuration settings for EC2 instances that you intend to launch repeatedly with consistency. To create a Launch Template, start by logging in to the AWS Management Console and navigating to the EC2 service. Once there, locate the "Launch Templates" section in the navigation pane and click on "Create launch template". Begin by specifying a name and version for your Launch Template, ensuring it reflects the purpose or configuration of the template. Then, proceed to configure the template settings. This includes specifying the Amazon Machine Image (AMI) to use for the instances, the instance type, key pair for SSH access, security groups, network settings, and other parameters.

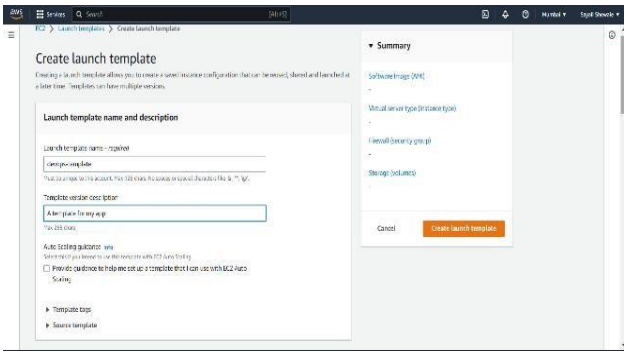


Fig-5 Launch Template

Implement Auto Scaling:

Creating an Auto Scaling Group (ASG) in Amazon Web Services (AWS) is a fundamental step in ensuring the availability and scalability of your EC2 instances to meet varying levels of demand. To create an ASG, begin

This work is licensed under a Creative Commons Attribution 4.0 International License

by logging in to the AWS Management Console and navigating to the EC2 service. In the EC2 dashboard, locate the "Auto Scaling Groups" section in the navigation pane and click on "Create Auto Scaling group".

Start by defining the launch configuration for your ASG, which specifies the AMI, instance type, key pair, security groups, and other settings for the EC2 instances launched by the ASG. Once the launch configuration is set up, configure the scaling policies for the ASG. These policies determine how the ASG will automatically adjust the number of instances based on metrics such as CPU utilization, network traffic, or custom CloudWatch metrics.

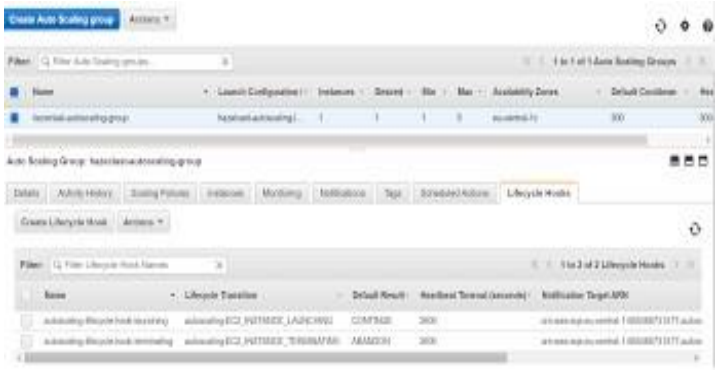


Fig-6 Auto Scaling Group

IV. RESULT

The server has been successfully installed and the Nginx startup page is displayed in the browser.The implementation of a load balancer and auto-scaling project yields a robust infrastructure capable of dynamically responding to fluctuations in traffic demand while ensuring high availability and optimal performance.

Fig-7 Server startup page

International Journal of Research in Engineering, IT and Social Sciences, ISSN 2250
6.565, Volume 09 Issue 03, June 2022, Page 4-12

V. CONCLUSION

In conclusion, the implementation of a load balancer and auto-scaling project offers significant benefits in terms of scalability, reliability, and cost-effectiveness for cloud-based applications. By leveraging load balancers,

This work is licensed under a Creative Commons Attribution 4.0 International License

incoming traffic is intelligently distributed across multiple instances, ensuring optimal resource utilization and enhanced performance. The integration of auto-scaling functionality further enhances the system's agility by automatically adjusting capacity in response to changing demand, thereby ensuring that the application remains responsive and available even during periods of peak usage. Additionally, the combination of these two components provides fault tolerance and resilience, with automatic instance replacement and load redistribution in the event of failures. Overall, a load balancer and auto-scaling project is essential for building scalable and highly available cloud infrastructures that can adapt to the dynamic nature of modern applications while optimizing operational costs.

VI. REFERENCES

[1] Danielo Gonc¸alves Gomes and Jos´e Neuman de Souza, "An Autonomic Computing-based Architecture for Cloud Computing Elasticity ," Virtual University Institute (UFC VIRTUAL) , Fortaleza - Brazil .IEEE,2015

[2] Satish Narayana Srirama, Alireza Ostovar, "Optimal Resource Provisioning for Scaling Enterprise Ap plications on the Cloud," Institute of Computer Science,University of Tartu, Estonia.IEEE,2015

[3] Shridhar G.Domanal and G. Ram Mohana Reddy, "Load Balancing in Cloud Environment using a Novel Hybrid Scheduling Algorithm ,"National Institute of Technology Karnataka, India.IEEE,2015 Surathkal, Mangalore

[4] Ying Liu, Navaneeth Rameshan, Enric Monte, Vladimir Vlassov and Leandro Navarro, "ProRenaTa: Proactive and Reactive Tuning to Scale a Distributed Storage System," KTH Royal Institute of Technology,Sweden.IEEE,2015

[5] Ali Yadavar Nikravesh, Samuel A. Ajila, Chung- Horng Lung, "Towards an Autonomic AutoScaling Prediction System for Cloud Resource Provisioning," Department of Systems andComputer Engineering,Carleton University 1125 Colonel By Drive, Ottawa K1S 5B6, Ontario Canada.IEEE,2015

[6] Sidra Aslam, Maunam Ali Shah, "Load Balancing in Cloud Computing: A Survey of Modern Techniques,"COMSATS Institute of information technology, Islamabad, Pakistan .IEEE,2015

[7] Mayanka Katyal, Atul Mishra, "A Comparative Study of Load Balancing in Cloud Computing Environment," YMCA University of Science and Technolohy ,Faridabad, Hariyana.India

[8] Sandeep Sharma, Sarabjit Singh, and Meenakshi Sharma, "Performance Analysis of Load Balancing Algorithms," World Academy of Science, Engineering and Technology.IEEE,2008

[9] A. Y. Nikravesh, S. A. Ajila, C.H. Lung, "Cloud resource autoscaling system based on Hidden Markov Model (HMM)",Proc. of the 8th IEEE International Conference on June 2014. Semantic Computing

[10] T. Lorido-Botran, J. Miguel-Alonso, J. A. Lozano, "A review of auto-scaling techniques for elastic applications in cloud environments," Journal of Grid Computing, vol. 12, no. 4, December 2014

[11] Resma K. S, Dr. Sharvani G. S, Edge Distributed Cloud Middleboxes International Journal Of Advance Research, Ideas And Innovations In Technology, ISSN: 2454-132X, Volume 5, Issue 3. Luo, J., Rao, L., & Liu, X. (2017). Spatio-Temporal Load Balancing For Energy Cost Optimization In Distributed Internet Data Centres. IEEE Transactions On Cloud Computing, 3(3), 387– 397.

This work is licensed under a Creative Commons Attribution 4.0 International License