**Knowledge Transfer (EC2 Cluster)**

**Dated : 27/07/2023**

**\*\*Challenge :\*\***

Hey geeks, today I want to give a task that you have to host a website using EC2 Instance and while the website is running on the browser it should not get downtime when the traffic on the site increases so manage the traffic and host it smoothly.

==================================================================================

**\*\*Task :\*\***

Ticket: Built a EC2 cluster for hosting website.

==================================================================================

**\*\*Solution :\*\***

Finally I have got the solution to host the site without any downtime using the EC2 inbuilt tools like Load balancers and Auto scaling. Now see this what this two tools will do which is unbelievable.

==================================================================================

**\*\*Pre-requisites:\*\***

• Have a good knowledge about EC2

• Know the all operations in EC2 instance

• Know tools about EC2

==================================================================================

**\*\*Objective:\*\***

To learn how to host a website without any issues.

==================================================================================

**\*\*Description:\*\***

An EC2 cluster is a collection of Amazon Elastic Compute Cloud (EC2) instances that work together to distribute the workload and improve the performance of applications. EC2 clusters are designed to handle large-scale computational tasks, parallel processing, and high-performance computing requirements. By grouping multiple EC2 instances into a cluster, organizations can leverage the combined processing power and memory of the instances to efficiently execute complex tasks, such as data analysis, simulations, and rendering.

A load balancer is a crucial component in modern cloud computing architectures, designed to evenly distribute incoming network traffic across multiple EC2 instances or virtual machines. This even distribution ensures that no single instance becomes overwhelmed by excessive traffic, thereby enhancing the overall availability and fault tolerance of applications. Load balancers monitor the health of each instance and intelligently route traffic, directing requests to the healthiest and least busy instances. This process optimizes resource utilization and prevents overloading, enabling efficient handling of sudden traffic spikes. Load balancers can be combined with auto scaling to dynamically adjust the number of instances based on traffic levels, further optimizing performance and cost efficiency. By distributing workloads across multiple instances and automatically adjusting capacity as needed, load balancers play a critical role in maintaining a stable and responsive application infrastructure in the cloud.

==================================================================================

**\*\*Steps:\*\***

**Step 1( Launching Instance )**

# Firstly Launch Linux Instance and allow the SSH and HTTP Permissions from Network Setting.

# Now launch the instance and type the following commands in it...

* sudo yum update
* sudo yum install httpd –y
* sudo systemctl start httpd
* sudo systemctl enable httpd

**Step 2( Downloading Winscp and Free-CSS Template )**

# Now Download Winscp on your windows/mac for tranfering files code( link = https://winscp.net/eng/index.php )

# Then download a free-css template for the website ( link = https://www.free-css.com ) and extract it

#After downloading do the following steps

* Open winscp application
* Provide the PublicIP Address #Available in Instance Summary
* Give Username ( ec2-user )
* Click on Advance for generating password
* Go to SSH and Authentication
* Then go to end of private key you will see a 3dot dialogue box click it
* Then click all files above the cancel button
* Then select the keypair of your instance and click ok
* Now click Login # After a remote server will open of your instance
* After this open the “/var/www/html/” directory in remote server
* Now copy the Free-CSS template files to remote server and save it

# If the prompt shows error or permission denied then go to linux shell and type

* sudo chmod -R 777 /var/www/html/

# now try again

# Finally Test the website by using PublicIP copy it and paste it in the browser

**Step 3( Creating Image )**

# Now the 3rd step we have to create Template image to create it follow the steps

* Select the Instance
* Go to actions
* Image and Template
* Create Image
* Give the name to your image
* Create Image

**Step 4( Launching Template )**

# For launching templates follow the steps

* Go to Launch Template
* Create launch template
* Give template name
* From Application and OS Images (Amazon Machine Image) select “currently in use”
* From Instance type select “Free tier eligible”
* From Key Pair select your currently used key pair or create new one
* From Network setting select “Create Security Group” # Do not use existing security group
* Give the name to security group
* Give the description to the security group
* Add Inbound Security Group Rules and in that add SSH and HTTP rules with source type “ANYWHERE”
* Now select Create Launch Template

**Step 5( Setting up Autoscale )**

# Now the 5th to set the automation to the instances depending upon the traffic on the site

**Step 1: Choose launch template or configuration**

* Go to Auto scaling group from the auto scaling section
* Then create auto scaling group
* Firstly give the auto scaling name
* Then in launch template select your template from the dropdown menu

**Step 2: Choose instance launch options**

# Select minimum 2 zones for HA and for backup if data is lost from 1 zone other should have backup

* From Availability Zones and subnets select zones

**Step 3: Configure advanced options**

* From Load Balancing section select “Attach to a new load balancer”
* In Attach to new load balancer select “Network Load Balancer”
* In Listeners and routing section from default routing select “Create a target group” from dialogue box

**Step 4: Configure group size and scaling policies**

* Select Desired Capacity = 2
* Select Minimum Capacity = 2

# We required minimum 2 instance because it should not directly go in downtime

* Select Maximum Capacity = 20

# We give Maximum 20 Instance for HA and smoothly working of the site without any problem even in high traffic

* Then in Scaling policies select “Target tracking scaling policy”
* Then in that give target value = 60 and Instances need = 200

# Target policies work on the CPU average utilization if the CPU use go to 60% alternate instance should be created within the instance need which is 200 second approx( 3mins ) for better HA

**Step 5: Add Notification**

# In this section you can give the option to notify you if the server is Launched, Terminate, Failed to launch, Failed to Terminate by the use of SNS service to your Email’s.

**Step 6: Add Tags**

# In this section add your tags if needed

**Step 7: Review**

# Before submitting please check all the things are correctly filled or not nothing should be left and click

* Create Auto Scaling

# Check in the Instances 2 Instances will be launched

**Step 6( Load Balancer )**

* Go to Load balancer in Load Balancing
* Select the DNS name
* Copy the DNS name and paste it in browser and see wheater the site is working or not

# If the site does not show in browser follow the steps

* Go to Load balancer
* Select it
* Go to actions and select “edit load balancer attributes”
* On the Cross-Zone load balancing and save it

# Now Re-Run the site with DNS name

=====================================================================================

**\*\*Explanation:\*\***

**1) Load Balancer:**

A load balancer acts as the traffic cop for incoming requests to a web application or service. When a user sends a request to access the application, the load balancer receives the request and then distributes it across multiple backend instances (e.g., EC2 instances) that are part of a target group. The load balancer uses various algorithms to intelligently determine which backend instance should handle each request. Some common algorithms include round-robin, least connections, and weighted distribution.

The load balancer constantly monitors the health of each backend instance by periodically sending health checks. If an instance becomes unhealthy or fails to respond, the load balancer automatically stops sending traffic to that instance until it becomes healthy again. This ensures that only healthy instances handle user requests, improving the overall availability and reliability of the application.

**2) Autoscaling:**

Autoscaling allows the automatic adjustment of the number of instances in an application's backend fleet based on predefined policies and metrics. When traffic to the application increases, and the existing instances start approaching their capacity limits, autoscaling kicks in to add more instances to the fleet. On the other hand, during periods of low traffic, autoscaling can remove unnecessary instances to reduce costs and resource wastage.

Autoscaling relies on predefined scaling policies that dictate when to scale up or down. These policies can be based on various metrics, such as CPU utilization, network traffic, or application-specific custom metrics. When a scaling event is triggered, autoscaling communicates with the cloud provider's infrastructure to provision or terminate instances as needed.

The integration of load balancers and autoscaling is crucial for maintaining a responsive and fault-tolerant application. As traffic increases, the load balancer distributes it across the available instances, while autoscaling dynamically adds new instances to the fleet to handle the increased load. Similarly, during low traffic periods, autoscaling can remove instances, and the load balancer redistributes the remaining traffic among the reduced number of instances. This dynamic and automated process ensures that the application can efficiently handle varying workloads while optimizing resource utilization and cost efficiency.