AWS AUTO SCALING

Creating group of EC2 instances that scale up or down depending on the conditions you set.

- Enable elasticity by scaling horizontally through adding or terminating EC2 instances.
- Auto scaling ensures that you have the right number of AWS EC2 instances for your needs at all time.
- Auto scaling helps you to save cost by cutting down the number of EC2 instances when not needed and scaling out to add more instances only it is required.

Auto Scaling Components:

1. Launch Configuration: like instance type, AMI, key-pair, security group
2. Auto Scaling Group: group name, group size, VPC, subnet, health check period
3. Scaling Policy: metric type, target value

How to Balance, Attach and Detach EC2 Instances:

Balance:

- If auto scaling finds that the number of EC2 instances launched by ASG into subjects AZs is not balanced (EC2 instances are not evenly distributed), auto scaling do rebalancing activity by itself.
- AS always tries to balance the instances distribution across AZs.
- While rebalancing, ASG launches new EC2 instances where there are less EC2 at present and then terminates the instances from the AZs that had more instances.

What causes imbalance of EC2?

- If we add or remove same subnets/AZ form auto scaling.
- If we manually request for EC2 termination from our ASG.
- An AZ that did not have enough EC2 capacity now has enough capacity and it is one of the auto scaling group.

Attach:

- We can attach a running EC2 instance to an ASG by using AWS console or CLI if the below conditions are meet:
    - Instances must be on running state.
    - AMI used to launch the EC2 still exists.
    - Instances is not the part of another auto scaling group.
    - Instances must be in the same AZ of the same group.

  - If the existing EC2 instances under the ASG, plus the one to be needed, exceeds the maximum capacity of the ASG, the request will fail, EC2 instance would not be added.

Detach:

- You can manually remove EC2 instances from an ASG using AWS console of CLI.
- You can then manage the detached instances independently or attach it to another ASG.
- When you detach an instance, you have the option to decrement the ASG desired capacity.
- If you do not, ASG will launch another instance to replace the one detached.
- When you delete an ASG its parameter like maximum, minimum and desired capacity are all set to zero. Hence it terminates it's all EC2 instances.
- If you want to keep the EC2 instances and manage them independently you can manually detach them first, then delete ASG.

- We can attach one more Elastic Load Balancer (ELB) to our ASG.
- The ELB must be in the same region as the ASG.
- Once you do this any EC2 instance existing or added by ASG will be automatically registered with the ASG defined ELB.
- Instances and the ELB must be in the same VPC.
- Auto scaling classifies its EC2 instance health check or unhealthy.
- By default, as uses EC2 status checks only to determine the health status of an instance.
- When you have one or more ELB defined with the ASG you can configure auto scaling to use both the EC2 status check and SLB health check to determine the instances health check.
- Health check grace period is 300sec by default.
- If we set zero in grace period, the instance health is checked once it is in service.
- Until the grace period timer expires any unhealthy status reported by EC22 status check of the ELB attached to the ASG will not be acted upon.
- After grace period expires ASG consider an instance unhealthy in any of the following cases:
  - EC2 status check report to ASG an instance other than running.
  - If ELB health check are configured to be used by the auto scaling, then if the ELB report the instance as 'out of service'.
- Unlike AZ rebalancing, termination of unhealthy instances happens first then auto scaling attempt to launch new instance to replace the ones terminated.
- Elastic IP and EBS volumes get detached from the terminated instances you need to manually attach there to the new instance.

Types of Auto Scaling Policies:

In four situations, ASG sends a SNS email notification:

- **i.** an instance is launched
- **ii.** an instance is terminated
- **iii.** an instance fails to launch
- **iv.** an instance fails to terminate

Merging ASG:

- Can only be done form the CLI not form the AWS console.
- You can merge multiple single AZ ASG into a single, one multi-AZ ASG.
- Scale out means launching more EC2 instances.
- Scale in means terminating one or more EC2 instances by scaling policy.
- It is always recommended to create a scale-in event for each scale-out event you create.
- AWS EC2 services sends EC2 metrics to CloudWatch about ASG instances.
- Basic monitoring is every 300sec enabled by default and free of cost.
- You can enable detailed every 60sec which is chargeable.
- When the launch configuration is done by AWS CLI, detailed monitoring for EC2 instances is enabled by default.

StandBy State:

- You can manually move an EC2 instance form an ASG and put it in standby state.
- Instances in standby state are still managed by auto scaling.
- Instances in standby state are charged as normal in service instances.
- They do not count towards available EC2 instances for workload/app use.
- Auto scaling does not perform health check on instance in standby state.

Scaling Policies:

Generally scaling policy is of two types such as:

1. Manual
2. Dynamic

Again, dynamic policy is divided into three categories as follows:

A. Target Tracking
B. Simple Scaling Policy
C. Step Scaling Policy

- Define how much you want to scale based on defined conditions.
- ASG uses alarms and policies to determine scaling.
- For Simple or Step scaling a scaling adjustment can't change the capacity of the group above the maximum group or below the minimum group.

Predictive/Scheduled/Cycle Scaling: it looks at historic pattern and forecast them into the future to schedule change in the number of EC2 instances. It uses machine learning model to forecast daily and weekly pattern.

Target Tracking Policies: increase or decrease the current capacity of the group based on a target value for specific metric. This is similar to the way that your thermostatic maintain the temperature of your home.

Step Scaling: increase or decrease the current capacity of the group based on a set of scaling adjustment known as step adjustment that vary based on the size of the alarm breach. It does not support/ wait for cool down times. It supports warm-up timer: time taken by newly launched instance to be ready and contribute to the watched metric.

Simple Scaling: single adjustment (up or down) in response to an alarm (cool down timer- 300sec by default)

Schedule Scaling: used for predictable load change. You need to configure a schedule action for a scale out at a specific date/time and to a required capacity. A scheduled action must have a unique data/time. You cannot configure two schedule activities at the same date/time.

ELASTIC LOAD BALANCER (ELB)

Load balancer distributes the web traffic to the available

server.

Or

Load balancing refers to efficient distributing incoming traffic across a group of backend server.

Load Balancer is of 3 types:

1. Classic Load Balancer
2. Application Load Balancer
3. Network Load Balancer

- An internet facing load balancer has a publicly resolvable DNS name.
- Domain names for content on the EC2 instances served by the ELB, is resolved by the internet DNS server to the ELB DNS name (and hence IP address).
- This is how traffic from the internet is directed to the ELB front-end.
- Classic load balancer service support: http, https, TCP, SSL.
  Protocols ports supported are 1-65535.
  It supports IPV4, IPV6 and Dual Stack.
- Application load balancer distributes incoming application traffic across multiple targets such as EC2 instances in multiple availability zone. This increases the availability of your application.
  - It supports protocols: HTTP/HTTPS (layer 7)
  - We can't assign elastic IP to ALB.
- Network load balancer has ability to handle volatile workloads and scale to millions of requests per seconds.
  - It supports protocols: TCP/UDP/TLC/SSL(layer 4)

- An ELB listener is the process that checks for connection request.
- You can configure the protocol/ port number on which your ELB listener listen for connection request.
- Fronted listeners check for traffic from client to the listener.
- Backend listeners are configured with protocol/port to check for traffic from the ELB to the EC2 instances.
- It may take some time for the registration of the EC2 instances under the ELB to complete.
- Registered EC2 instances are those are defined under the ELB.
- ELB has nothing to do with the outbound traffic that is initiated/generated from the registered EC2 instances destined to the internet or to any other instances within the VPC.
- ELB only has to do with inbound traffic destined to the EC2 registered instances (as the destination) and the respective return traffic.
- You start to be charged hourly (also for partial hours) once your ELB is active.
- If you do not want to be charged as you so not need the ELB anymore, you can delete it.
- Before you delete the ELB, it is recommended that you point the Route53 to somewhere else other than ELB.
- Deleting the ELB does not affect or delete the EC2 instance registered with it.

- ELB forwards traffic to “eth0” of your registered instances.
- In case the EC2 registered instances has multiple IP address on eth0, ELB will route the traffic to its primary IP address.
- Elastic load balancer supports IPV4 address only in VPC.
- To ensure that the ELB service can scale ELB nodes in each AZ, ensure that the subnet defined for the load balancer is at least /27 in scale size and has at least 8 available IP address the ELB nodes can use to scale.
- For fault tolerance it is recommended that you distribute your registered EC2 instances across multiple AZ with in the VPC region.
- If possible, try to allocate same number of registered instances in each AZ.
- The load balancer also monitors the health of its registered instances and ensures that it routes traffic only to healthy instances.
- A healthy instance shows as healthy under the ELB.
- When the ELB detects an unhealthy instance, it stops routing traffic to that instance.
- An unhealthy instance shows as unhealthy under the ELB.
- By default, AWS console uses ping http (port 80) for healthy check.
- Registered instances must respond with an http “200 OK” message within the timeout period else it will be considered as unhealthy.

- AWS API uses ping TCP (port-80) for health check.
- Response time-out is 5 seconds (range is 2-60 sec).
- Health check internet.
- Period between health check (default 30 and range is 5 to 300 sec)

- **Unhealthy Threshold:** number of consecutive failed health check that should occur before the instance is declared unhealthy.
  Range is 2 to 10
  Default is 2

- **Healthy Threshold:** number of consecutive successful health checks that must occur before the instance considered unhealthy.
  Range is 2 to 10
  Default is 10

- By default, the ELB distributes traffic evenly between the AZ, it is defined in without consideration to the number of registered EC2 instances in each AZ.

Cross Zone Load Balancing:

- Disabled by default.
- When enabled, the ELB will distribute traffic evenly between registered EC2 instances.
- If you have 7 EC2 instances in one AZ, and 3 in another AZ, and you enabled cross zone
  Load balancing each registered EC2 instances will be getting the same amount of traffic load from the ELB.
- ELB name you choose must be unique within the account.
- ELB is region specific, so all registered EC2 instances must be in the same region but can be in different AZs.
- To define your ELB in an AZ you can select one subnet in that AZ. Subnet can be public or private.
- Only one subnet can be defined for the ELB in an AZ.
- If you try and select another one in the same AZ, it will replace the former one.
- If you register instance in an AZ with ELB but do not define a subnet in that AZ for the ELB, these instances will not receive traffic form the ELB.
- ELB should always be accessed using DNS and not IP.

An ELB can be internet facing or internal ELB:

- **Internet Facing:**
  - ELB nodes will have public IP address.
  - DNS will resolve the ELB DNS name to these IP address.
  - If routes traffic to the private IP address of your registered EC2 instances.
  - You need one public subnet in each AZ where the internet facing ELB will be defined such that the ELB will be able to route internet traffic.

- Format of the public ELB DNS name of internet facing ELB:

  name-1234567890.region.elb.amazonaws.com

- Format of the internal ELB:

  Internal-name.123456789.region.elb.amazonaws.com

- An ELB listener is the process that checks for connection request.
- Each network load balancer needs at least one listener to accept traffic.
- You must assign a security group to your ELB. This will control traffic that can reach your ELB front end listeners.

Target Group:

- Logical grouping of targets behind the load balancer.
- Target groups can be existing independently from the load balancer.
- Target group can be associated with an auto scaling group.
- Target group can contain up to 200 targets.

- Clients connect to the **listener** of the load balancer.
- The load balancer connects to one or more **targets** or servers.
- Two connections in play:

- Listener connection: one connection between the client and listener.
- Backend connection: one connection between load balancer and target.

- The LB abstracts the client away from individual servers.

- Used for high availability, fault tolerance, and scaling. Targets are one single compute resource that connections are connected towards.
- Target groups are groups of targets which are addressed using rules.
- Rules are
  - path based
  - host based if you want to use different DNS names.
- Support EC2, EKS, Lambda, HTTPS, HTTP/2 and websockets.
- ALB can use SNI for multiple SSL certs attached to that LB.
  - LB can direct individual domain names using SSL certs at different target groups.
- AWS does not suggest using Classic Load Balancer (CLB), these are legacy.
  - This can only use one SSL certificate.