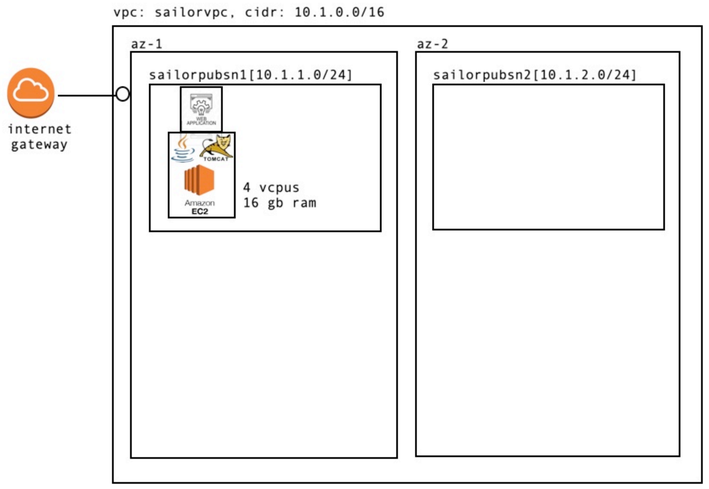


Here we are deploying only one-instance of our application and running which could lead to below problems.
1. since we are running only one-single instance of the application it could lead to single-point of failure (not highly-available). If the single-instance of the application crashes, the whole system will goes down.

2. scalability: if huge traffic comes to the application, the single instance may not be able to handle.

one solution to address the scalability while running single-instance of the application is using scale-up/scale-down

#1. scale-up & scale-down
scale-up & scale-down is modifying the shape of the resource
Incase of scale-up the shape of the resource like cpu, memory, storage etc will be increased. Incase of scale-down these computing resources will be decreased.

looks like for addressing the huge traffic towards the application we can use one of the solution as scale-up & scale-down but it has lot of limitations as described below.
1. when we scale-up/scale-down the resources the cloudplatform will restart the resources to have these changes being reflected. so that the application will be down and will not become accessible which causes loss of service to the business
2. even though we can increase the cpu, memory and storage resources to these compute instances, there is always an max or upper limit towards them. once we reach to the limit we are blocked and cannot make changes.
3. where there is an increase in traffic towards the application, it not only demands more cpu or computing resources, to handle such a huge traffic we need more network i/o capacity, which cannot be addressed with scale-up and scale-down
4. due to heavy network traffic towards the resource, there is huge latency in serving the requests

From the above scale-up and scale-down is not the right solution for handling huge traffic towards the application.

Where and When we should use scale-up and scale-down?

developers - build the application
to run the application we need infrastructure and computing resources (ec2 instance).

How do we determine what is the shape of the instance to be used in running the application?
we should not endup creating the compute instance with random shape or higher shape as it results in huge billing cost. there are several aspects taken into consideration in determing the shape of instance to be used in running application

1. operating system platform = the operating system vendor provides specifications interms of resources required for running it
2. platform software being used to run the application = for eg.. a tomcat server provider, gives us the specifications interms of resources required for running on a machine
3. software application = nature of the application that has been built.

taking all these into account, the business has to determine what is the average traffic towards the application at anypoint of time. based on that create an virtual load to determine how much computing capacity is being used to identify the shape of the machine required for running the application.
taking anything more than the above derived form of the shape would endup in wastage of the resources and high cost.

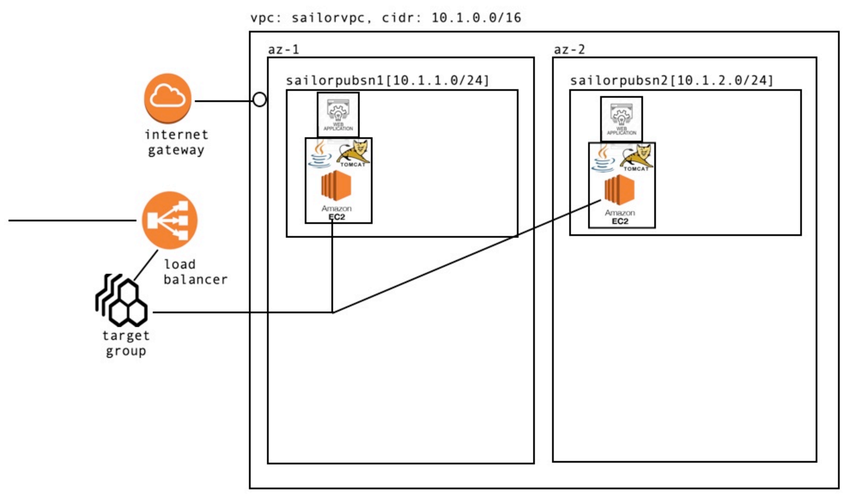
If the developers has comeup with an newer release of the software application that may contains:
1. change in technology stack
2. upgrade of the underlying platform softwares
3. addition of new modules or features to the application
that demands the change in the computing resources required in running the newer version.

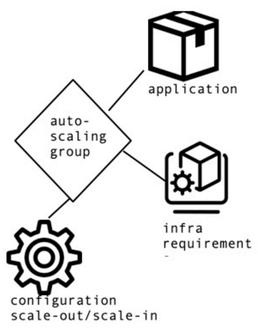
In this case rather than setting up the new compute instance with high shape required, we can use scale-up in extending the computing capacity of the exiting instance that has minimal effect in rolling the application.

To address the problem with
1. single-point of failure
2. scalability

solution:
we need to always run multiple instances of the application across the availability zones of the region and distribute the traffic across these instances using loadbalancer. since we have multiple instances running, there is always “High-Availability”

What about scalability?
since we #2 instances of the application we can distribute traffic across these instances to handle the user traffic.





if the traffic increases further, where it cannot be handled by only #2 nodes, then what we should do?
We should add more nodes to the targetgroup distributing the traffic across:
1. provision one more ec2 instance
2. install required software packages on it like jdk, tomcat etc
3. perform necessary configurations to deploy the application
4. deploy the software application onto that newly created instance and run it
5. add the new instance as a node into the target group, so that once the healthcheck has been passed, it will be scheduled or routed to accept the traffic

The amount of time it takes in provisioning, installing, configuring, deploying/running the application and adding it as node to the targetgroup to route the traffic is very huge and by the time the application instances that are running might crash.

So how to address the above problem of scaling up the application dynamically inorder to handle huge traffic loads: ASG is the solution.

#1.