**Re-factoring with AWS**

1. **Intro:**

Instead of using auto-scaling groups and tomcat instances etc..,things we used in the previous project, we are going to use aws managed services and some paas and saas on aws cloud.

1. **Scenario:**

Let’s take the same scenario as the previous project, when we had some running services on physical/virtual/cloud machines(EC2), and we are dealing with various services that requires various teams. So we will be using a cloud platform, and instead of using Iaas(ec2), we will be using mostly saas & paas services that are very easy to manage; flexible and elastic in nature. Scaling will be mostly taken care of by aws.

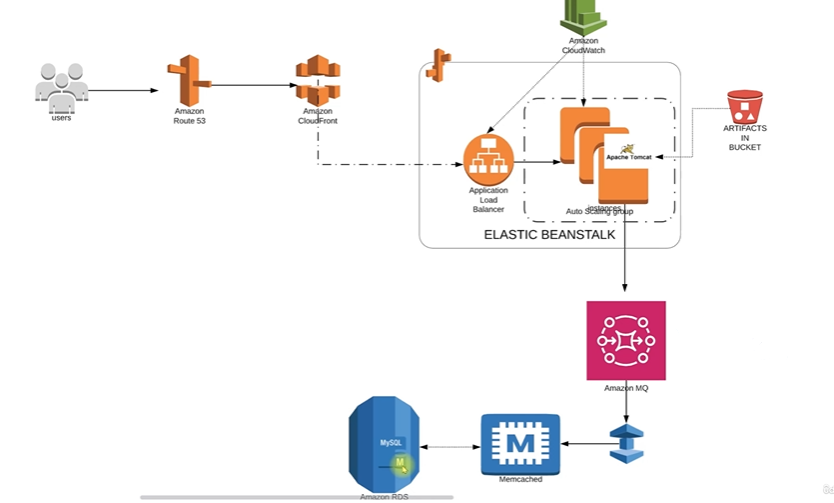
1. **Why?**

Re-factoring our apps in aws will give us an easy infrastructure to manage, very good performance and very convenient to scale.

1. **AWS services:**

* Front-end
* Beanstalk => VM for tomcat
* Nginx LB replacement
* Automation for VM scaling
* EFS/S3 bucket for storing the artifacts
* Back-end
* RDS instance for DB( Paas) instead of mySQL
* Elastic Cache instead of memcached
* Active MQ instead of RabbitMQ
* Route 53 for DNS
* CloudFront for CDN to serve the global audience

1. **Architecture**

****

1. **Flow of execution:**

* -Login to aws acc
* - Create key pair for beanstalk instance login
* - Create security group for Elastic Cache, RDS,Active MQ
* - Create Elastic Cache, RDS,Active MQ
* - Create Elastic Beanstalk environment
* - Update SG of backend to allow traffic from Bean SG
* - Update SG of backedn to allow internal traffic
* - Launch EC2 instance for DB initializing
* - Login to the instance and initialize RDS DB
* - Change healthcheck on Bean to login
* - Add 443 https listener to LB
* - Build artifact from source code
* - Deploy artifact to Beanstalk
* - Create SDN with ssl cert

1. **Implementation:**

**-Step1: Create sg for backend services**

**-Step2: Create RDS**

First create subnet in which we are going to create our RDS instance.Then configure parameter group that contains informations about which db you want.After that create RDS DB and insert all this parameter into it.

**-Step3: Create elastiCache**

Configure parameter group that contains informations about ELC. After that create subnet group and finally create memcached cluster and insert all previous parameters in it.

**-Step4: Create Active MQ**

We have two choices: apache ActiveMQ or RabbitMQ. Choose RabbitMQ and single instance broker in one availability zone

**-Step5: Initialize our RDS instance**

First copy rds endpoint, then go to launch an ec2 instance and log in to it. Install mysql by bootstrapping the vm, then connect to db by typing mysql –h “your endpoint” –u “user name” –p “password and show db you will find the db parameter you created in aws RDS.

-**Step6: AWS elastic Beanstalk**

Now when all our backend services are ready, we collect all our services’s informations: RDS user and password and endpoint, Activemq user and password and endpoint and elastiCach endpoint.

Now let’s create a beanstalk environment:think of it as a big container to run multiple environments.

Basically beanstalk gives instances and load balancer, so we need to configure our instance inside it.

We should also edit capacity for setting up type of instances,type of scaling ..

And the most important thing is configuring rolling updates and deployments:

This option gives us a choice to when we deploy our artifact, how do we want it to be deployed: Rolling is better for deployment policies for this use case.

After that we will modify security for adding key pairs to log in to the ec2 instance that will be launched by Beanstalk.

If the first environment you created gets terminated, that’s because you should add a service role.

Create a new env with the same settings and associate a service role in the security section.

**-Step7: Modify backend secGroup and update**

Now that our env is ready, we should consider adding RDS, memcached and activeMQ services as inbound rules in the backend secG. We should also consider linking ssl certificate, configure health..

**-Step8: Deploy our artifact**

Time to deploy our app from source code and deploy to Beanstalk env.

From a political side, you should have maven and jdk installed on your machine.

So first, we should update the rds, memc, and rabbit endpoints in the app.properties file, give user and pass for mysql.

Now we deploy our artifact by typing mvn install which will give us the artifact that will be uploaded to beanstalk env.

**-Step9: Setup cloudfront**

**You can be specific in cdn to give global access by choosing to distribute the app in edge locations**