DevOps Assignments

Pre-requisite for Running the Script.

1 . Terraform is an infrastructure as code software should be installed on your system.

2 . Install Git on your system.

Application Environment used:

1. Apache for static content

2. Tomcat for dynamic content

[https://github.com/nitishNec/data](https://github.com/nitishNec/dataS)

[S](https://github.com/nitishNec/dataS)tep to run the script for infra provisioning.

Run the git clone command to take the checkout of github repo

>>git clone https://github.com/nitishNec/data

In data folder there are various file and folder present

1. **main.tf** (All the resources are defined in this file )

2. **outputs.tf** (All the output that we want after creating the infra like ec2 public ip ,loadbalancer dns etc are defined in this file )

3.**variable.tf** (All the variables that are used in creating the infra are defined in this section )

Above all three file are mainly for creating the infra using terraform

4.**web.sh** (This file has shell script called in terrafrom for configuration of system like installation of Apache , Git , mod\_ssl, java , tomcat etc and this shell script will take the checkout of github repo which consist of companyNews.war and static content to deploy on the server also this shell script call another shell script **ssl.sh** to generate the self signed certificateon server )

5.**ssl.sh. (**This shell script will generate the ssl certificate on the server to run the application on HTTPS  **)**

**6.ssl.conf (**Used in Apache for hosting the application on https  **)**

**7.non\_ssl.conf (**Used in apache for redirecting all the http request to https **)**

First generate/create the access key and secret key in your AWS account.

Go to the data directory using cd command.

Place access\_key and secret\_key value in main.tf file in provider section of main.tf

access\_key = “”

secret\_key =””

After doing the above run the below commands

Run the terraform command

1. Terraform init

2. Terraform plan

3. Terraform apply

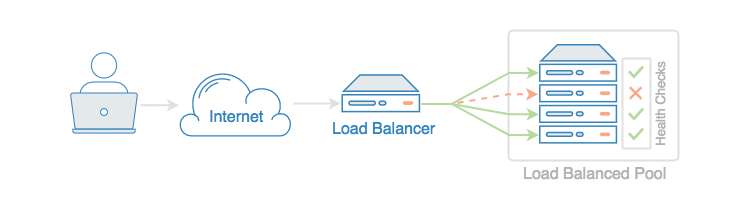
After doing that infrastructure will be created .

Infra consist of Ec2 instances ,loadbalancer,Autosacling,Security group with ingrees and egress rules for ec2 ,Security group for loadbalancer and application like tomcat , apache, etc will be installed and our war file and static content willalso be deployed.

In this Assignment i am using the autoscaling , load balancer service of AWS for scaling the public release as well as targeting 4-9’s availability of this application .

AWS Auto Scaling is a service that automatically monitors and adjusts compute resources to maintain performance for applications hosted in the Amazon Web Services public cloud.

AWS Auto Scaling service is best for an application that uses one or more scalable resources and experiences variable load. A good example would be an e-commerce web application that receives variable traffic through the day. It follows a standard three tier architecture with Elastic Load Balancing for distributing incoming traffic,



Elastic Load Balancing automatically distributes incoming application traffic across multiple targets, such as Amazon EC2 instances It can handle the varying load of your application traffic in a single Availability Zone or across multiple Availability Zones.

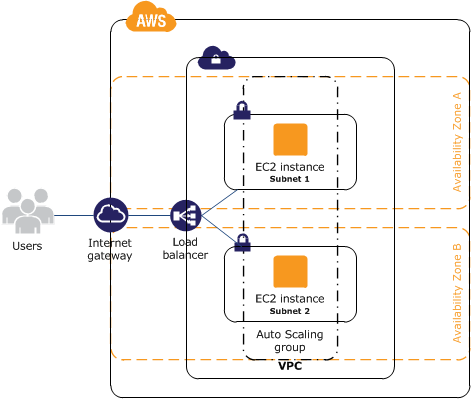


Diagram 2

Considering the Diagram 2 .

Using the Terraform I have created the **1 load Balancer ,1 Autoscaling group with minimun 2 instance in auto scaling group and maximum number of instance in Auto Scaling group is 10 .**

On one of the ec2 instance i have deploy the static file on Apache server and on the other instance i have put the dynamic content .i.e comanyNews.war file on tomcat server.

So for the limited Release it will works fine but **when we go for the fully public (production/live) then based on traffic our auto scaling will launch up to 10 aws instance with same configuration** .

Also if we need more infra then we will change the value of min ,desired and max number instance in auto scaling configuration of terraform template (main.tf ) and run the Terraform apply .

Also as i have created the load balancer this lb will evenly distribute traffic on our ec2 instance(our app ) created in auto scaling .

Once our infra is fully ready after running the terraform we need to check the https request either using p**ublic DNS of load balancer** or it can also be checked by hitting the hosted domain **www.exale.com** .

Once you hit above url in browser it will serve your page on **https** as i have also redirected all the **http(port 80) traffic to https (port 443 ) that uses the SSL certificate** .

Taking into the account of Release we can use the Blue-Green deployment So that we can achieve the 4-9’s availability during release also .