**Action plan:** Installing ruby on rails and with integrating apache with phusion passenger webserver, we can deploy the ruby app. We can set autoscaling in the AWS server so that the load can be balanced. The installation and configuration can be automated using chef cookbook.

**Steps:**

**Step1:**

First step I did was to set up an amazon cloud instance. Setting up an amazon instance can be done with the help of console. I have chosen Ubuntu instance to do the job.

**Step2:**

Updating the installed packages by using the following command.

sudo apt-get update

**step3:**

Making a directory ruby in the user home directory.

mkdir ~/ruby

navigating into the directory.

cd ~/ruby

**step4:**

Getting the ruby verison using wget.

wget http://cache.ruby-lang.org/pub/ruby/2.1/ruby-2.1.4.tar.gz

inflating the tar.

tar -xzf ruby-2.1.4.tar.gz

**step5:**

Navigating into the ruby directory.

cd ruby-2.1.4

running the configure file will check for the dependencies and it will create a make file, which contains all that need to compile a code.

./configure

**step6:**

Running the make utility makes the makefile executable.

make

Running the make utility with the install parameter copies the binaries into the /usr/local/bin directory.

sudo make install

Now the ruby is installed.

**Step7:**

Installing apache2 webserver.

sudo apt-get install apache2

Updating installed packages

sudo apt-get update

**Step8:**

installing passanger package

**Step9:**

sudo apt-get install libapache2-mod-passenger

Enabling passanger apache module

sudo a2enmod passenger

restarting apache2

sudo service apache2 restart

**Step10:**

installing rails gem.

cd ~

sudo gem install --no-rdoc --no-ri rails

creating a new app

rails new app --skip-bundle

editing gemfile in the app. And inserting gem 'sinatra'

gem 'puma'

cd app

nano Gemfile

bundle install

**Step10:**

creating the virtual host for the apache for the app. This can be done by copying the default configuration file of apache2.

sudo cp /etc/apache2/sites-available/000-default.conf /etc/apache2/sites-available/app.conf

editing the conf file and adding the document root.

sudo nano /etc/apache2/sites-available/app.conf

<VirtualHost \*:80>

ServerAdmin webmaster@localhost

DocumentRoot /home/rails/testapp/public

RailsEnv development

ErrorLog ${APACHE\_LOG\_DIR}/error.log

CustomLog ${APACHE\_LOG\_DIR}/access.log combined

<Directory "/home/rails/testapp/public">

Options FollowSymLinks

Require all granted

</Directory>

</VirtualHost>

Disabling the default apache site and enabling the site with app in it and restarting the webserver.

sudo a2dissite 000-default

sudo a2ensite app

sudo service apache2 restart

Thus, the application is deployed in the AWS server and the public DNS will be,

http://ec2-52-90-113-255.compute-1.amazonaws.com./

**Step11: Making the application scalable.**

I have configured an auto scaling group to the Existing AWS instance using console. So that it automatically scales the load coming to the server.

**Step12: Infrastructure automation.**

I am automating the infrastructure using Chef. So I am installing chef in the workstation and using the hosted chef server and automating the installation process in the node.

**Step13:**

Installing chef on the workstation.

We can download the chef package from the chef website and transfer the zip file onto the workstation.

Unzip the file and we will get an chef development kit and the chef starter kit file.

Install the .deb file by using the command

Sudo dpkp -I chef dk\_0. 9. 0-1 amd64.deb

Then we have to create an chef repo in the root directory.

Mkdir chef-repo

In the chef-repo we can unzip the starter kit.

Unzip chef-starter-kit.zip

This will enable the connection between workstation and the hosted chef server.

Now in the chef repo we can generate cookbooks with the recipes in it.

We can download the cookbooks and apache from the chef supermarket, I have used two cookbooks from the chef super market, which are installing ruby, installing apache2 and passenger.

We can do it by using command,

knife cookbook site install ruby\_install

knife cookbook site install apache2

knife cookbook site install passenger

After downloading cookbooks, we have to make a runlist in the hosted chef by pushing them

Knife cookbook upload ruby\_install

Knife cookbook upload apache2

Knife cookbook upload passenger.

Now, we have all the cookbooks in a runlist. When ever we need to spin an another server using these packages, we can just bootstrap the node by using the IP address.

Knife bootstrap 10.0.2.15 --ssh-user manoj --ssh-password --sudo --use-sudo-password --node -name node1 --runlist ‘recipe[ruby\_install, apache2, passenger]’