Installation of Rocket chat Aws:

Setting up the Aws environment:

-->First set up an Aws account on Amazon web services.

-->After setting it up go to **“open Ec2 service** ”,click on Instances to the left side of your Ec2 dashboard and select an Ec2 instance.

-->Now go to the search bar and search for “**Ubuntu Server 18.04 LTS**” with 64 -bit (x86) Architecture.

-->Select your instance(t2 -medium) and click **“Next: Configure Instance** “

-->Keep it default in the storage or choose your storage according to your usage.

-->In this section set everything as default.click “Next “

-->This is Security Group add HTTP and HTTPS using their drop downs.click on “**Review and Launch**”

-->This leads to creation of a key pair, now select **generate a new key pair** and name the key pair as you like, keep the key pair secure since the key pair allows you to securely SSH into your instance. Finally “**Click on launch Instances**”.

Configuring Elastic IP:

-->Now click on view instances and you can see the instance running.

-->On your Ec2 dashboard you can see a section called Elastic iP click on that.

→ click “**Allocate new Address**”. Click “**Allocate**”.

→ Copy the Ip address and save it. On the top find the **Actions button**, click on it and select **Associate address**.

--> Now you can notice a drop down Instance and private Ip select what it shows .(if you have multiple then see which instances are running followed by their ip ).

Configuring Route 53:

-->Buy a domain either from aws or any other domain hosting company

--> Go to create hosted zone on dashboard of route 53

-->Create Record set

--> type : **Cname**, value: **Public Dns** .

Connecting to your instance:

-->Go to your Instance and click connect, go to SSH client and follow the steps.

--> open your terminal(linux) or putty for windows.

-->Now connect to your server using **chmod 400 “Your key”** this command is executable only if your in the same directory of your key present.

--> SSH the command, copy and paste it in your terminal and say yes for connecting.

-->Congratulations you are logged into your server.

Deployment Rocket chat:

Getting an SSL Certificate:

-->First type **sudo su** for root user

-->Installing certbot with the following commands **sudo apt update**

**sudo apt install certbot**

-->To get your SSL certificate from “letsencrypt” use:

**sudo certbot certonly --standalone --email xxxxx@gmail.com -d mydomain.com**

So, **etc/letsencrypt/archive** creates 4 pem files

Cert.pem

Fullchain.pem------(admins certificate file)

Chain.pem

Privkey.pem---------(certificate key file)

Configure Nginx web server :

-->Now **sudo apt-get Install nginx**

-->Enable it by **sudo ufw enable** and say yes to it

--> Lets check the status by **sudo ufw status** and see the status : Active

-->**sudo ufw app list** gives list of available applications

-->Now if you want all then **sudo ufw allow ‘nginx full’** ( for individual application change with ‘nginx name\_app’)

-->Type **sudo ufw status** to see available applications along with where and action.

-->Type **systemctl status nginx** this function shows green light enabled and is active for both webserver and reverse proxy. This statement conveys that the ip address of our ipv4 has started working.

-->Check by typing ipv4 address on any browser on our computer and you can see it.

--> Backup the default config file using **cd /etc/nginx/sites-available** followed with **sudo mv default default.reference**

-->Creating a new site configuration for Rocket.Chat

**sudo nano etc/nginx/sites-available/default** which will open a blank file and paste the following given below

server {

listen 443 ssl;

server\_name mydomain.com;

ssl\_certificate /etc/letsencrypt/live/mydomain.com/fullchain.pem;

ssl\_certificate\_key /etc/letsencrypt/live/mydomain.com/privkey.pem;

ssl\_protocols TLSv1 TLSv1.1 TLSv1.2;

ssl\_prefer\_server\_ciphers on;

ssl\_ciphers 'EECDH+AESGCM:EDH+AESGCM:AES256+EECDH:AES256+EDH';

root /usr/share/nginx/html;

index index.html index.htm;

# Make site accessible from http://localhost/

server\_name localhost;

location / {

proxy\_pass http://localhost:3000/;

proxy\_http\_version 1.1;

proxy\_set\_header Upgrade $http\_upgrade;

proxy\_set\_header Connection "upgrade";

proxy\_set\_header Host $http\_host;

proxy\_set\_header X-Real-IP $remote\_addr;

proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;

proxy\_set\_header X-Forwarded-Proto http;

proxy\_set\_header X-Nginx-Proxy true;

proxy\_redirect off;

}

}

server {

listen 80;

server\_name mydomain.com;

return 301 https://$host$request\_uri;

}

-->Here mydomain.com is my domain name. Save this file

-->Test the Nginx **sudo nginx -t** where you will get a message : **etc/nginx/nginx.conf syntax ok and test is successful**

Install GIT:

->**sudo apt install git**

-->**git --version**

-->**git clone** <https://github.com/RocketChat/Docker.Official.Image.git>

Install Docker:

-->**sudo apt-get update**

-->**sudo systemctl start docker**

-->**sudo systemctl enable docker**

-->verify the installation **sudo systemctl status docker**

Install Docker compose:

--> **sudo curl -L "https://github.com/docker/compose/releases/download/1.26.0/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose**

--> **sudo chmod +x /usr/local/bin/docker-compose**

Setting up docker containers:

--> **sudo mkdir -p /opt/docker/rocket.chat/data/runtime/db**

-->**sudo mkdir -p /opt/docker/rocket.chat/data/dump**

After these commands create a docker-compose.yml

--> **sudo nano /opt/docker/rocket.chat/docker-compose.yml**

Copy this and paste it:

version: '2'

services:

rocketchat:

image: rocket.chat:latest

command: >

bash -c

"for i in `seq 1 30`; do

node main.js &&

s=$$? && break || s=$$?;

echo \"Tried $$i times. Waiting 5 secs...\";

sleep 5;

done; (exit $$s)"

restart: unless-stopped

volumes:

- ./uploads:/app/uploads

environment:

- PORT=3000

- ROOT\_URL=https://mydomain.com

- MONGO\_URL=mongodb://mongo:27017/rocketchat

- MONGO\_OPLOG\_URL=mongodb://mongo:27017/local

depends\_on:

- mongo

ports:

- 3000:3000

mongo:

image: mongo:4.0

restart: unless-stopped

command: mongod --smallfiles --oplogSize 128 --replSet rs0 --storageEngine=mmapv1

volumes:

- ./data/runtime/db:/data/db

- ./data/dump:/dump

# this container's job is just to run the command to initialize the replica set.

mongo-init-replica:

image: mongo:4.0

command: >

bash -c

"for i in `seq 1 30`; do

mongo mongo/rocketchat --eval \"

rs.initiate({

\_id: 'rs0',

members: [ { \_id: 0, host: 'localhost:27017' } ]})\" &&

s=$$? && break || s=$$?;

echo \"Tried $$i times. Waiting 5 secs...\";

sleep 5;

done; (exit $$s)"

depends\_on:

- mongo

After the above Start the container by

-->**cd /opt/docker/rocket.chat**

-->**sudo docker-compose up -d**

The above two will start rocket chat and mongo db

Individual Containers:

First, start an instance of mongo:

-->**docker run --name db -d mongo:4.0 mongod --smallfiles**

Then start Rocket.Chat linked to this mongo instance:

-->**docker run --name rocketchat --link db:db -d rocket.chat**

This will start a Rocket.Chat instance listening on the default Meteor port of 3000 on the container.If you'd like to access the instance directly at standard port on the host machine:

-->**docker run --name rocketchat -p 80:3000 --env ROOT\_URL=http://mydomain --link db:db -d rocket.chat**

Then, access it via http://localhost in a browser. Replace localhost in ROOT\_URL with your own domain name if you are hosting at your own domain.

**Final:**

Login to your site at <https://mydomain.com> ---it is finally deployed on aws.

How to remove the deployment and cleanup all resources created for it:

-->Remove Docker(images,containers and volumes):

**sudo apt-get autoremove -y --purge docker-engine docker docker.io docker-ce**

The above command will not remove images, containers, volumes, or user created configuration files on your host. If you wish to delete all images, containers, and volumes run the following commands:

**sudo rm -rf /var/lib/docker /etc/docker**

**sudo rm /etc/apparmor.d/docker sudo groupdel docker**

**sudo rm -rf /var/run/docker.sock**

-->Remove Git :

**sudo apt-get remove --auto-remove** (git all dependency packages too)

To delete configuration and/or data files of git and it's dependencies execute:**$ sudo apt-get purge --auto-remove git**

-->Remove Ngnix:

**sudo apt purge nginx**

This commands deletes all configurations .

-->Remove Certbot:

**sudo certbot delete**

Remove certbot files manually

**sudo rm -rf /etc/letsencrypt/**

**sudo rm -rf /var/lib/letsencrypt/**

**sudo rm -rf /var/log/letsencrypt/**

Make sure the repo is updated and autoremove

**sudo apt update**

**sudo apt upgrade**

**sudo apt autoremove**

These steps can clean up everything that was created.

**Additional methods:**

-->Remove Docker compose:

sudo rm $(which docker-compose)

-->Remove all docker containers:

Clean Docker and start from scratch, enter the command:

**docker container stop $(docker container ls –aq) && docker system prune –af ––volumes**