Universidad de San Carlos de Guatemala

Facultad de Ingeniería

Escuela de Ciencias y Sistemas

Sistemas Operativos 1

Proyecto de Suficiencia

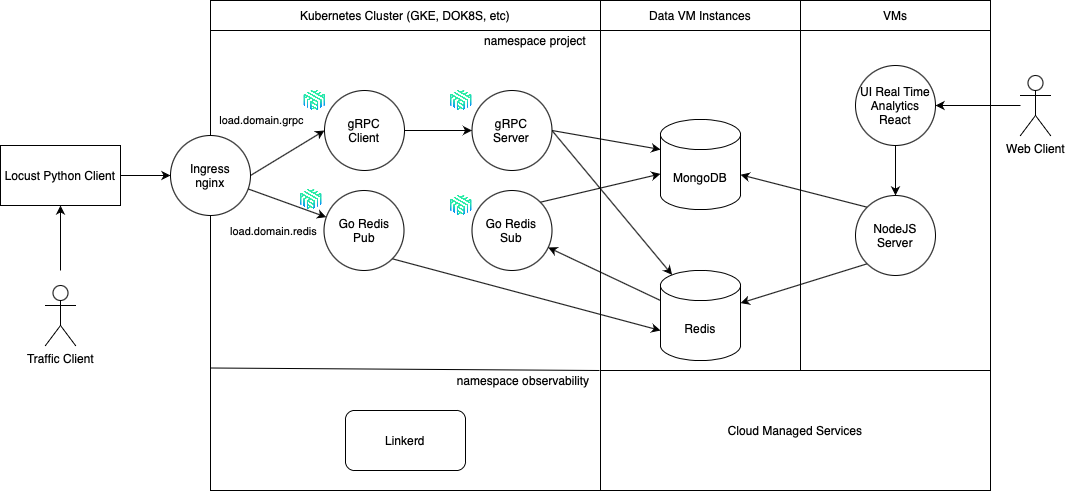
Manual técnico

Andrea María López Flores

201404134

Guatemala, 4 de febrero de 2021

# Arquitectura de la aplicación



# Descripción

La aplicación está compuesta por un cliente el cual fue desarrollados con Python para tener generar tráfico en la aplicación, esto fue creado por medio de Locust. Los datos recibidos pasan a un balanceador de carga creado con Ingress nginx, el cual envía los datos al servicio de gRPC client y a Regis Pub los cuales luego los enviarán a los otros servicios que a su vez almacenarán la información en una base de Redis y una base de Mongo. La información almacenada es consumida por una aplicación de NoseJS ya cual realiza diferentes consultas para que estas sean consumidas por React que mostrará gráficas de las consultas realizadas por el servicio de NodeJS.

Se creó la imagen Docker de cada uno de los nodos de la aplicación, fueron subidos a Docker hub y ejecutadas en una maquina virtual de Google Cloud.

Cada uno de los nodos de la aplicación fue creado como Deployments de Kubernetes los cuales están inyectados con Linkerd para ser monitoreados. Y luego se creó un servicio por cada uno de ellos. Todo esto dentro de un namespce llamado Project.

# Comandos de configuración

Los siguientes comandos fueron utilizados para configurar las imágenes, crear los deployments, crear servicios, consultar objetos del namespace, configurar locust, configurar linkerd.

COMANDOS REDIS

sudo apt update

sudo apt install redis-server

sudo nano /etc/redis/redis.conf

supervised systemd

sudo systemctl restart redis.service

sudo systemctl status redis

sudo systemctl disable redis

redis-cli

COMANDOS MONGO

https://www.digitalocean.com/community/tutorials/how-to-install-mongodb-on-ubuntu-18-04-source

CREAR IMAGEN DOCKER

sudo docker login

sudo docker build -t <nombre> .

sudo docker tag <nombre> andreaflores/<nombre>

sudo docker push andreaflores/<nombre>

-------------------

sudo docker login

sudo docker build -t servergrpc .

sudo docker tag servergrpc andreaflores/servergrpc

sudo docker push andreaflores/servergrpc

sudo docker login

sudo docker build -t clientgrpc .

sudo docker tag clientgrpc andreaflores/clientgrpc

sudo docker push andreaflores/clientgrpc

sudo docker login

sudo docker build -t servernode .

sudo docker tag servernode andreaflores/servernode

sudo docker push andreaflores/servernode

sudo docker login

sudo docker build -t reactui .

sudo docker tag reactui andreaflores/reactui

sudo docker push andreaflores/reactui

DETENER Y BORRAR IMAGEN

sudo docker stop 40b227e09f02 9de06616ef68

sudo docker rm 40b227e09f02

INICIAR IMAGEN

kubectl run clientgrpc --image=clientgrpc

CREAR DEPLOYMENT

kubectl create deployment clientgrpc --image=andreaflores/clientgrpc -n project

kubectl get deployments -n project

kubectl get pods -n project

MOSTRAR NODOS

kubectl get nodes -o wide

CREAR SERVICIOS

kubectl expose deployment servergrpc --port=50051 --target-port=50051 --type=ClusterIP -n project

kubectl expose deployment app1 --port=80 --target-port=80 --type=NodePort

kubectl expose deployment redispub --port=5000 --target-port=5000 --type=LoadBalancer -n project

BORRAR SERVICIOS

kubectl delete service <nombre>

MOSTRAR OBJETOS DE UN NAMESPACE

kubectl get services -n project

kubectl get deployments -n project

BORRAR OBJETOS DE UN NAMESPACE

kubectl delete deployment clientgrpc -n project

IMAGENES CREADAS DE DOCKER

sudo docker images

LINKERD

kubectl get -n project deploy -o yaml \

| linkerd inject - \

| kubectl apply -f –

https://linkerd.io/2/getting-started/

IR A LINKERD

curl -sL https://run.linkerd.io/install | sh

export PATH=$PATH:/c/Users/Andrea/.linkerd2/bin

linkerd dashboard

http://localhost:50750/

DOMINIOS

grpc.suficiencia.34.68.118.118.xip.io

redis.suficiencia.34.68.118.118.xip.io

CARGAR LOCUST

# En la carpeta de locust

locust -f carga.py

CONEXION A LA MAQUINA VIRTUAL

ssh -i id\_rsa Andrea@35.223.38.239

ABRIR ARCHIVO INGRESS NGINX

nano ingress.yaml

IP DE SERVIDOR DE NODEJS

http://104.197.230.46:3001/all

IP DE REACT

<http://35.226.217.95:3000/edad>

VER DATOS EN REDIS

SUBSCRIBE casos

keys \*