Distributed Systems - Mini-project

Icesi University
Subject: Distributed Systems
Professor: Daniel Barragán C.
Topic: ***Kubernetes***

Students:	Alejandro Bueno	- A00335472
	Rubén Ceballos	- A00054636
	Nicolás Recalde	- A00065888
	Ana Valderrama	- A00065868

Git URL: https://github.com/abc1196/sd2018b-project/tree/abueno/sd2018b-project

Goals

- ✓ Identify the components involved in a *kubernetes* cluster.
- ✓ Deploy a *kubernetes* cluster.
- ✓ Deploy applications in a *kubernetes* cluster using its properties of persistent volumes, load balancing and service discovery.
- ✓ Diagnose and execute autonomously the necessary actions to achieve stable infrastructures.

Prerequisites

- Kubernetes Cluster

Description

For this project we are going to deploy a *kubernetes* cluster with 3 nodes, one master and two workers. For that, we are going to use Google Cloud Platform that provides a lot of benefits like: load balancing, automatic scaling, node pools, etc. To achieve this is necessary to follow a series of steps, provided in an easy Google guide:

- https://google.qwiklabs.com/focuses/878?locale=en&parent=catalog&qlcampaign=77-18-gcpd-236&utm_source=gcp&utm_medium=documentation

To demonstrate the correct functioning of the cluster we are going to deploy a pod executing a web service. The deployment will have 3 replicas and each pod will have a label. This application could be accessed through the browser.

Development

The goals of this project were achieved using **Google Cloud Platform**. First step is to *log in* to **Google Cloud Console**, if you have a GCP account is preferred to use the one provided for qwiklabs to not generate charges.

Once we are in, it is necessary to activate **Google Cloud Shell** to access to command-line. When the environment is provisioned and connected, we can start. This shell is a virtual machine that has all the tools we will need.

The project on Google Cloud is already created, so we have to set the default compute zone, and then it is ready to create the cluster.

```
gcloud config set compute/zone us-central1-a
```

A cluster consists of at least one machine, which corresponds to the master, and multiple worker machines called nodes. To create the cluster we run the following command:

```
gcloud container clusters create sd2018b-project
```

Where sd2018b-project corresponds to the name of the cluster. It is really simple, after the cluster is created. The credentials to access to it are needed so, we get it with the following command:

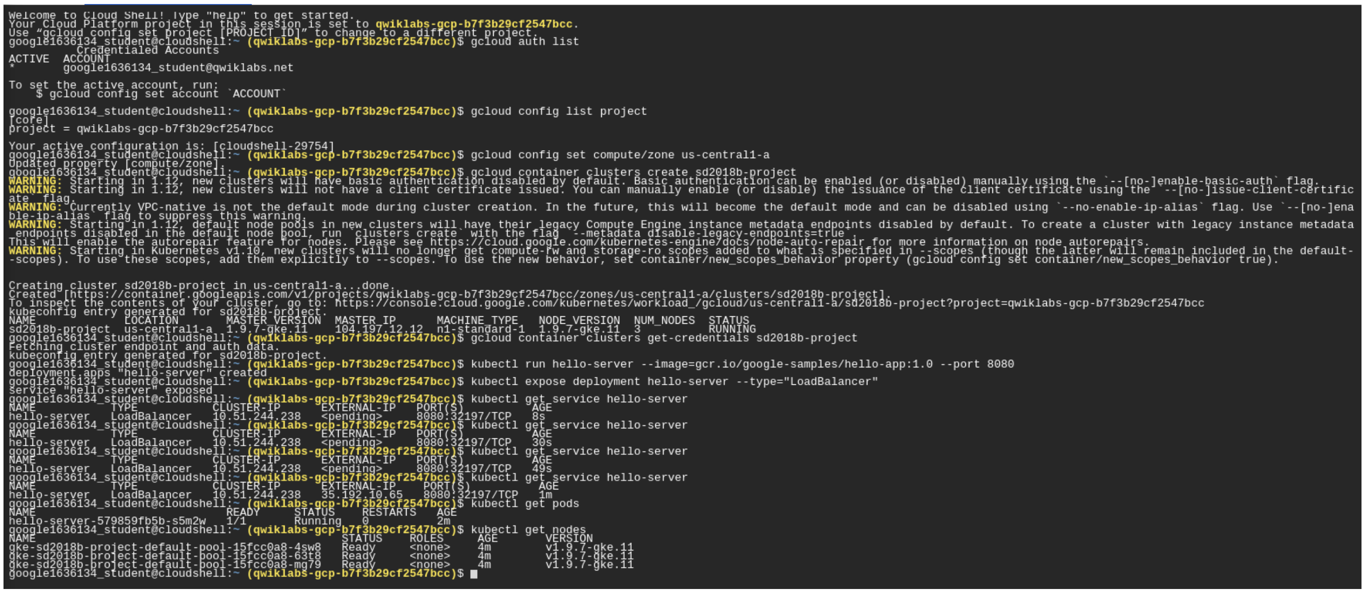
```
gcloud container clusters get-credentials sd2018b-project
```

Now, we are going to deploy a web application that will be accessible through the browser. We run the application on the cluster using a sample image provided by Google and we defined a port that container will expose.

```
kubectl run hello-server --image=gcr.io/google-samples/hello-app:1.0 --port 8080
```

Once we deploy the app, we created a kubernetes service to handle the external traffic, ***load balancer***, this create 2 additional nodes to handle requests (3 in total).

```
kubectl expose deployment hello-server --type="LoadBalancer"
```

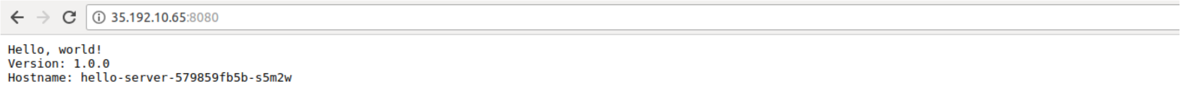


```
Welcome to Cloud Shell! Type "help" to get started.
Your Cloud Platform project in this session is set to qwiklabs-gcp-b7f3b29cf2547bcc.
Use “gcloud config set project [PROJECT_ID]” to change to a different project.
google1636134_student@cloudshell:~ (qwiklabs-gcp-b7f3b29cf2547bcc)$ gcloud auth list
          Credentialed Accounts
ACTIVE  ACCOUNT
*       google1636134_student@qwiklabs.net

To set the active account, run:
    $ gcloud config set account `ACCOUNT`

google1636134_student@cloudshell:~ (qwiklabs-gcp-b7f3b29cf2547bcc)$ gcloud config list project
[core]
project = qwiklabs-gcp-b7f3b29cf2547bcc

Your active configuration is: [cloudshell-29754]
google1636134_student@cloudshell:~ (qwiklabs-gcp-b7f3b29cf2547bcc)$ gcloud config set compute/zone us-central1-a
Updated property [compute/zone].
google1636134_student@cloudshell:~ (qwiklabs-gcp-b7f3b29cf2547bcc)$ gcloud container clusters create sd2018b-project
WARNING: Starting in 1.12, new clusters will have basic authentication disabled by default. Basic authentication can be enabled (or disabled) manually using the `--[no-]enable-basic-auth` flag.
WARNING: Starting in 1.12, new clusters will not have a client certificate issued. You can manually enable (or disable) the issuance of the client certificate using the `--[no-]issue-client-certific
ate` flag.
WARNING: Currently VPC-native is not the default mode during cluster creation. In the future, this will become the default mode and can be disabled using `--no-enable-ip-alias` flag. Use `--[no-]ena
ble-ip-alias` flag to suppress this warning.
WARNING: Starting in 1.12, default node pools in new clusters will have their legacy Compute Engine instance metadata endpoints disabled by default. To create a cluster with legacy instance metadata
 endpoints disabled in the default node pool, run `clusters create` with the flag `--metadata disable-legacy-endpoints=true`.
This will enable the autorepair feature for nodes. Please see https://cloud.google.com/kubernetes-engine/docs/node-auto-repair for more information on node autorepairs.
WARNING: Starting in Kubernetes v1.10, new clusters will no longer get compute-rw and storage-ro scopes added to what is specified in --scopes (though the latter will remain included in the default-
-scopes). To use these scopes, add them explicitly to --scopes. To use the new behavior, set container/new_scopes_behavior property (gcloud config set container/new_scopes_behavior true).


Creating cluster sd2018b-project in us-central1-a...done.
Created [https://container.googleapis.com/v1/projects/qwiklabs-gcp-b7f3b29cf2547bcc/zones/us-central1-a/clusters/sd2018b-project].
To inspect the contents of your cluster, go to: https://console.cloud.google.com/kubernetes/workload_/gcloud/us-central1-a/sd2018b-project?project=qwiklabs-gcp-b7f3b29cf2547bcc
kubeconfig entry generated for sd2018b-project.
NAME             LOCATION       MASTER_VERSION  MASTER_IP       MACHINE_TYPE   NODE_VERSION   NUM_NODES  STATUS
sd2018b-project  us-central1-a  1.9.7-gke.11    104.197.12.12   n1-standard-1  1.9.7-gke.11   3          RUNNING
google1636134_student@cloudshell:~ (qwiklabs-gcp-b7f3b29cf2547bcc)$ gcloud container clusters get-credentials sd2018b-project
Fetching cluster endpoint and auth data.
kubeconfig entry generated for sd2018b-project.
google1636134_student@cloudshell:~ (qwiklabs-gcp-b7f3b29cf2547bcc)$ kubectl run hello-server --image=gcr.io/google-samples/hello-app:1.0 --port 8080
deployment.apps "hello-server" created
google1636134_student@cloudshell:~ (qwiklabs-gcp-b7f3b29cf2547bcc)$ kubectl expose deployment hello-server --type="LoadBalancer"
service "hello-server" exposed
google1636134_student@cloudshell:~ (qwiklabs-gcp-b7f3b29cf2547bcc)$ kubectl get service hello-server
NAME           TYPE           CLUSTER-IP      EXTERNAL-IP   PORT(S)          AGE
hello-server   LoadBalancer   10.51.244.238   <pending>     8080:32197/TCP   8s
google1636134_student@cloudshell:~ (qwiklabs-gcp-b7f3b29cf2547bcc)$ kubectl get service hello-server
NAME           TYPE           CLUSTER-IP      EXTERNAL-IP   PORT(S)          AGE
hello-server   LoadBalancer   10.51.244.238   <pending>     8080:32197/TCP   30s
google1636134_student@cloudshell:~ (qwiklabs-gcp-b7f3b29cf2547bcc)$ kubectl get service hello-server
NAME           TYPE           CLUSTER-IP      EXTERNAL-IP   PORT(S)          AGE
hello-server   LoadBalancer   10.51.244.238   <pending>     8080:32197/TCP   49s
google1636134_student@cloudshell:~ (qwiklabs-gcp-b7f3b29cf2547bcc)$ kubectl get service hello-server
NAME           TYPE           CLUSTER-IP      EXTERNAL-IP    PORT(S)          AGE
hello-server   LoadBalancer   10.51.244.238   35.192.10.65   8080:32197/TCP   1m
google1636134_student@cloudshell:~ (qwiklabs-gcp-b7f3b29cf2547bcc)$ kubectl get pods
NAME                           READY     STATUS    RESTARTS   AGE
hello-server-579859fb5b-s5m2w  1/1       Running   0          2m
google1636134_student@cloudshell:~ (qwiklabs-gcp-b7f3b29cf2547bcc)$ kubectl get nodes
NAME                                             STATUS    ROLES     AGE       VERSION
gke-sd2018b-project-default-pool-15fcc0a8-4sw8   Ready     <none>    4m        v1.9.7-gke.11
gke-sd2018b-project-default-pool-15fcc0a8-63t8   Ready     <none>    4m        v1.9.7-gke.11
gke-sd2018b-project-default-pool-15fcc0a8-mg79   Ready     <none>    4m        v1.9.7-gke.11
google1636134_student@cloudshell:~ (qwiklabs-gcp-b7f3b29cf2547bcc)$
```

Figure 1. Kubernetes cluster deployed with web server application and load balancing running.

Demonstration

Once the nodes are running in the cluster, we can access to the web browser, and check that the web application is ok.

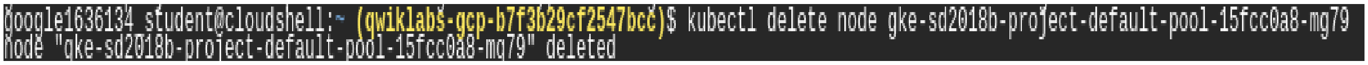


35.192.10.65:8080

```
Hello, world!
Version: 1.0.0
Hostname: hello-server-579859fb5b-s5m2w
```

Figure 2. Web application running.

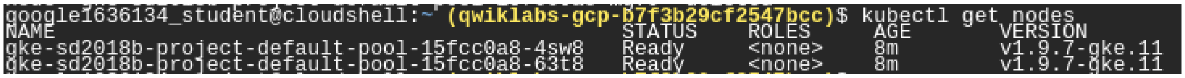
Afterwards, we deleted a node to see if the application continued running, we executed the following command:



```
google1636134_student@cloudshell:~ (qwiklabs-gcp-b7f3b29cf2547bcc)$ kubectl delete node gke-sd2018b-project-default-pool-15fcc0a8-mg79
node "gke-sd2018b-project-default-pool-15fcc0a8-mg79" deleted
```

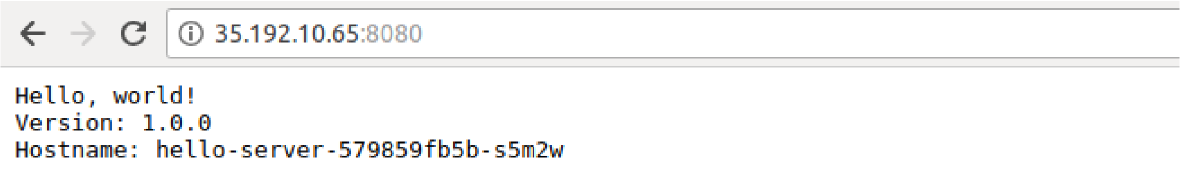
Figure 3. Command executed to delete a node and message showed.

We appreciate only two nodes on console, so we went check the web service and it continued working.



```
google1636134_student@cloudshell:~ (qwiklabs-gcp-b7f3b29cf2547bcc)$ kubectl get nodes
NAME                                             STATUS    ROLES     AGE       VERSION
gke-sd2018b-project-default-pool-15fcc0a8-4sw8   Ready     <none>    8m        v1.9.7-gke.11
gke-sd2018b-project-default-pool-15fcc0a8-63t8   Ready     <none>    8m        v1.9.7-gke.11
```

Figure 4. Active nodes.



```
Hello, world!
Version: 1.0.0
Hostname: hello-server-579859fb5b-s5m2w
```

Figure 5. Web service running with only two nodes.

References

- https://google.qwiklabs.com/focuses/878?locale=en&parent=catalog&qlcampaign=77-18-gcpd-236&utm_source=gcp&utm_medium=documentation