

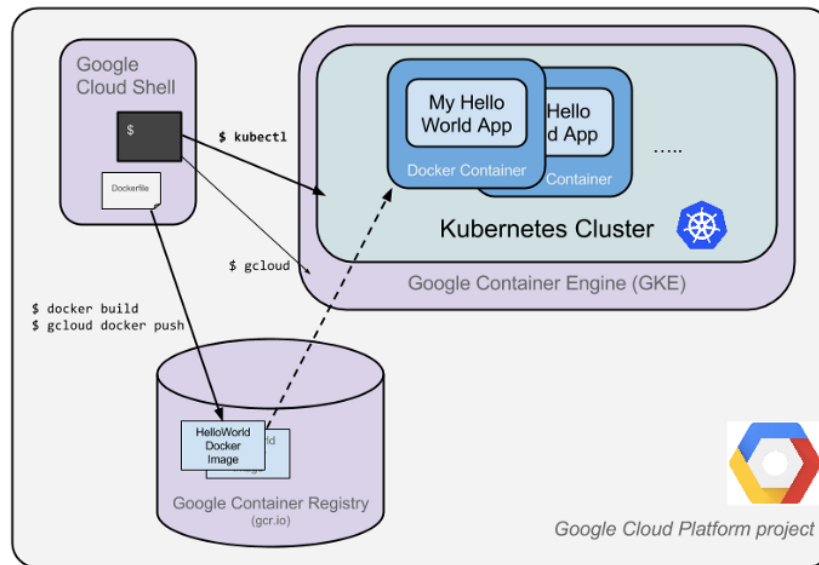
GCP Kubernetes in four steps



Prashanta Paudel

Nov 30, 2018 · 2 min read

This blog will demonstrate how to deploy containers using kubernetes engine in GCP.



How kubernetes work

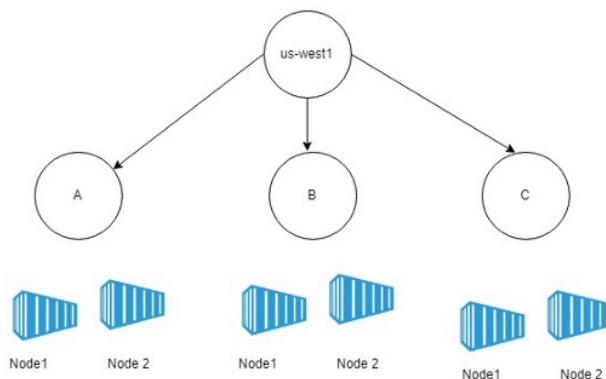
Step one: Build a cluster



STEP ONE > build a cluster

```
$ gcloud container clusters create web-server --machine-type=n1-standard-1 --num-nodes=2 --region=us-west1
```

| NAME | LOCATION | MASTER VERSION | MASTER IP | MACHINE TYPE | NODE VERSION | NUM_NODES | STATUS |
|------------|----------|----------------|----------------|---------------|--------------|-----------|---------|
| web-server | us-west1 | 1.9.7-gke.11 | 35.233.165.159 | n1-standard-1 | 1.9.7-gke.11 | 6 | RUNNING |



Upto Now nothing is loaded into cluster , only cluster is built.

Step 2: Build a workload/Container

I am going to build a small hello world website for test purpose but it could be anything from a thousand lines of codes.

To built a docker image we need code and docket file. So, I created a folder containing index file and docker file.

Lastly, I created a docker image from these two files.

STEP TWO > build a workload/Container

```
prashantagcp@cloudshell:~$ cd kubetest
prashantagcp@cloudshell:~/kubetest$ nano index.html
prashantagcp@cloudshell:~/kubetest$ ls
prashantagcp@cloudshell:~/kubetest$ nano Dockerfile
prashantagcp@cloudshell:~/kubetest$ ls
Dockerfile index.html
```

```
<!DOCTYPE html>
<html>
<body>
<h1>My First Heading</h1>
<p>My first paragraph.           Hello world !</p>
</body>
</html>
```

```
FROM centos:latest
CMD ["yum install apache2"]
COPY . /var/www/html
```

```
prashantagcp@cloudshell:~/kubetest$ docker build -t website-image:v1 .
Sending build context to Docker daemon 3.072kB
Step 1/3 : FROM centos:latest
latest: Pulling from library/centos
aeb7866da422: Pull complete
Digest: sha256:dc29e2bccceac52af0f01300402f5e756cc8c44a310867f6b94f5f7271d4f3fec
Status: Downloaded newer image for centos:latest
--> 75835a67d134
Step 2/3 : CMD ["yum install apache2"]
--> Running in 9a95a4c7a533
Removing intermediate container 9a95a4c7a533
--> 33e4417ff7f6
Step 3/3 : COPY . /var/www/html
--> 05e3f264dc62
Successfully built 05e3f264dc62
Successfully tagged website-image:v1
```

Step 3: Upload to Google Container Registry

After the container is built it needs to be uploaded to GCP for implementing.

Command

```
$ gcloud builds submit --tag gcr.io/project_name/website-image:v1 .
```

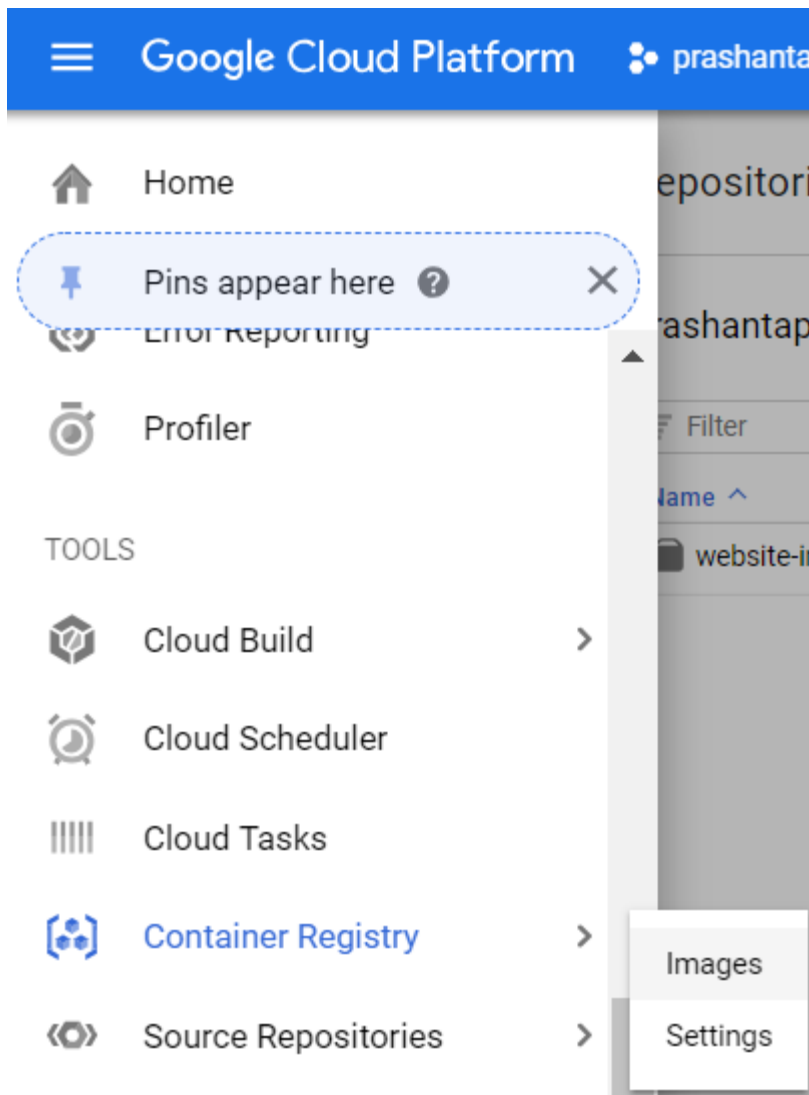
```

prashantap@cloudshell:~/code$ gcloud builds submit --tag gcr.io/prashantapaudel-777/website-image:v1 .
Creating temporary tarball archive of 2 file(s) totalling 197 bytes before compression.
Uploading tarball of [...] to [gcr.io/prashantapaudel-777/cloudBuild/source/1543546655.97-2232414473e84235c989514b9274e80.tgz]
API [cloudbuild.googleapis.com] not enabled on project [724960580787].
Would you like to enable and retry (this will take a few minutes)?
(y/N)? y
Enabling service [cloudbuild.googleapis.com] on project [724960580787]...
Waiting for service operation operations/acf.1392524-2e8d4d52-b1a4-c08cf74de1e to complete...
Operation finished successfully. The following command can describe the operation details:
gcloud services operations describe operations/acf.1392524-2e8d4d52-b1a4-c08cf74de1e
Created [https://cloudbuild.googleapis.com/v1/projects/prashantapaudel-777/builds/th3b8d44-3432-474c-9771-9727e893c1ee].
Logs are available at [https://console.cloud.google.com/gcr/builds/th3b8d44-3432-474c-9771-9727e893c1ee?project=724960580787].
-----
starting build "th3b8d44-3432-474c-9771-9727e893c1ee"
-----
FETCHSOURCE
Fetching source object: gcr.io/prashantapaudel-777/cloudBuild/source/1543546655.97-2232414473e84235c989514b9274e80.tgz#1543546655e615220
Copying gcr.io/prashantapaudel-777/cloudBuild/source/1543546655.97-2232414473e84235c989514b9274e80.tgz#1543546655e615220...
/ [1 file(s)] 316.0 B/ 316.0 B
Operation completed over 1 object(s)/316.0 B.
BUILD
Already have image (with digest): gcr.io/cloud-builders/docker
Sending build context to Docker daemon 3.072kB
Step 1/3 : FROM centos:latest
latest: Pulling from library/centos
Digest: sha256:dc9e7eb0e3c1a0421fe75d0c0c44a310867f4b94f1f7271d4ef3fec
Status: Downloaded newer image for centos:latest
--> 7833b6c7d814
Step 2/3 : CMD ["yum install apache2"]
--> Running in 2ff3b994e6d4
Removing intermediate container 2ff3b994e6d4
--> b42258f2c7e4
Step 3/3 : CMD /usr/sbin/httpd
--> 499ab4d19bc
Successfully built 499ab4d19bc
Successfully tagged gcr.io/prashantapaudel-777/website-image:v1
Docker
Pushing gcr.io/prashantapaudel-777/website-image:v1
The push refers to repository [gcr.io/prashantapaudel-777/website-image]
b4a371eeb60: Preparing
f972d139738d: Preparing
f972d139738d: Layer already exists
9d4a37bee60: Pushed
v1 digest: sha256:ae797cf6c9c1d0253034edf923e41f62a0c0d17c4509973e7476e616737f535 size: 736
DONE
-----

```

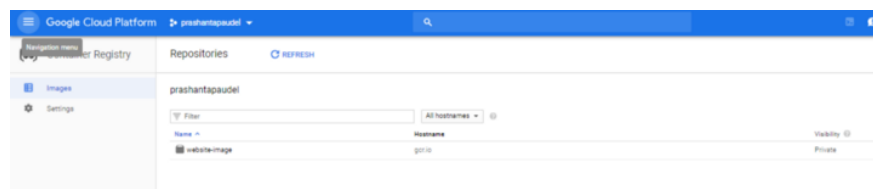
uploading to container registry

Let's check whether it is in registry or not



Go to container registry > images

Hurray! we have the image now.



Images

Step four: Deploy from Registry

After you have the image in the registry, just deploy to the cluster we have built in step one.

```
$ kubectl run website --image=gcr.io/project name/website-  
image:v2 --port 8080  
  
$ kubectl expose deployment website --type=Loadbalance --  
port=80 --target-port=8080
```

