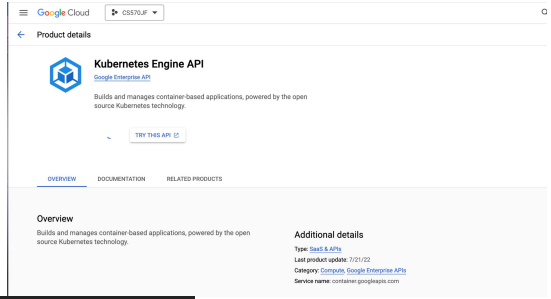
PySpark on Kubernetes

Word Count+ PageRank

Jisen Fang

Implementation-Create a Kubernetes cluster

1. Enable Kubernetes Engine API
2. gcloud container clusters create w7h1 --num-nodes=1 --machine-type=e2-highmem-2 --region=us-west2




```
Welcome to Cloud Shell! Type "help" to get started.
Your Cloud Platform project in this session is set to cs570jf.
Use "gcloud config set project [PROJECT_ID]" to change to a different project.
jfang757@cloudshell:~ (cs570jf)$ gcloud container clusters create w7h1 --num-nodes=1 --machine-type=e2-highmem-2 --region=us-west2
Default change: VPC-native is the default mode during cluster creation for versions greater than 1.21.0-gke.1500. To create advanced routes based clusters, please pass the `--no-enable-ip-alias` flag
Default change: During creation of nodepools or autoscaling configuration changes for cluster versions greater than 1.24.1-gke.800 a default location policy is applied. For Spot and PVM it defaults to ANY, an
 for all other VM kinds a BALANCED policy is used. To change the default values use the `--location-policy` flag.
Note: Your Pod address range (`--cluster-ipv4-cidr`) can accommodate at most 1008 node(s).
Creating cluster w7h1 in us-west2... Cluster is being health-checked (master is healthy)...done.
Created [https://container.googleapis.com/v1/projects/cs570jf/zones/us-west2/clusters/w7h1].
To inspect the contents of your cluster, go to: https://console.cloud.google.com/kubernetes/workload_/gcloud/us-west2/w7h1?project=cs570jf
kubeconfig entry generated for w7h1.
NAME: w7h1
LOCATION: us-west2
MASTER_VERSION: 1.26.5-gke.1200
MASTER_IP: 34.102.59.207
MACHINE_TYPE: e2-highmem-2
NODE_VERSION: 1.26.5-gke.1200
NUM_NODES: 3
STATUS: RUNNING
jfang757@cloudshell:~ (cs570jf)$
```

gcloud container clusters create w7h1 --num-nodes=1 --machine-type=e2-highmem-2 --region=us-west2

```
Welcome to Cloud Shell! Type "help" to get started.
Your Cloud Platform project in this session is set to cs570jf.
Use "gcloud config set project [PROJECT_ID]" to change to a different project.
jfang7578@cloudshell:~ (cs570jf)$ gcloud container clusters create w7h1 --num-nodes=1 --machine-type=e2-highmem-2 --region=us-west2
Default change: VPC-native is the default mode during cluster creation for versions greater than 1.21.0-gke.1500. To create advanced routes based clusters, please pass the `--no-enable-ip-alias` flag
Default change: During creation of nodepools or autoscaling configuration changes for cluster versions greater than 1.24.1-gke.800 a default location policy is applied. For Spot and PVM it defaults to ANY, an
 for all other VM kinds a BALANCED policy is used. To change the default values use the `--location-policy` flag.
Note: Your Pod address range (`--cluster-ipv4-cidr`) can accommodate at most 1008 node(s).
Creating cluster w7h1 in us-west2... Cluster is being health-checked (master is healthy)...done.
Created [https://container.googleapis.com/v1/projects/cs570jf/zones/us-west2/clusters/w7h1].
To inspect the contents of your cluster, go to: https://console.cloud.google.com/kubernetes/workload_/gcloud/us-west2/w7h1?project=cs570jf
kubeconfig entry generated for w7h1.
NAME: w7h1
LOCATION: us-west2
MASTER_VERSION: 1.26.5-gke.1200
MASTER_IP: 34.102.59.207
MACHINE_TYPE: e2-highmem-2
NODE_VERSION: 1.26.5-gke.1200
NUM_NODES: 3
STATUS: RUNNING
jfang7578@cloudshell:~ (cs570jf)$
```

Create image and deploy spark to Kubernetes

1. Install the NFS Server Provisioner
   helm repo add stable https://charts.helm.sh/stable
   helm repo update

```
jfang757@cloudshell:~ (cs570jf)$ helm repo update
Hang tight while we grab the latest from your chart repositories...
...Successfully got an update from the "stable" chart repository
Update Complete. ⎈Happy Helming!⎈
jfang757@cloudshell:~ (cs570jf)$
```

   helm install nfs stable/nfs-server-provisioner --set persistence.enabled=true,persistence.size=5Gi

2. Create a persistent disk volume and a pod to use NFS
   spark-pvc.yaml
   vim spark-pvc.yaml

3. Apply the yaml descriptor
   kubectl apply -f spark-pvc.yaml

```
jfang757@cloudshell:~ (cs570jf)$ kubectl apply -f spark-pvc.yaml
persistentvolumeclaim/spark-data-pvc created
pod/spark-data-pod created
jfang757@cloudshell:~ (cs570jf)$
```

```
jfang757@cloudshell:~ (cs570jf)$ vim spark-pvc.yaml
jfang757@cloudshell:~ (cs570jf)$ cat spark-pvc.yaml
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: spark-data-pvc
spec:
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: 2Gi
  storageClassName: nfs
---
apiVersion: v1
kind: Pod
metadata:
  name: spark-data-pod
spec:
  volumes:
    - name: spark-data-pv
      persistentVolumeClaim:
        claimName: spark-data-pvc
  containers:
    - name: inspector
      image: bitnami/minideb
      command:
        - sleep
        - infinity
      volumeMounts:
        - mountPath: "/data"
          name: spark-data-pv
jfang757@cloudshell:~ (cs570jf)$
```

4. Create and prepare your application JAR file
   docker run -v /tmp:/tmp -it bitnami/spark -- find /opt/bitnami/spark/examples/jars/ -name spark-examples* -exec cp {} /tmp/my.jar \;

```
jfang757@cloudshell:~ (cs570jf)$ docker run -v /tmp:/tmp -it bitnami/spark -- find /opt/bitnami/spark/examples/jars/ -name spark-examples* -exec cp {} /tmp/my.jar \;
Unable to find image 'bitnami/spark:latest' locally
latest: Pulling from bitnami/spark
0e0346ffa270: Pull complete
Digest: sha256:46f6fc4f1db377a71ec1866b340dfe47ae511ddff7b94ce4066e8582ae884c2f
Status: Downloaded newer image for bitnami/spark:latest
spark 04:51:39.77
spark 04:51:39.77 Welcome to the Bitnami spark container
spark 04:51:39.77 Subscribe to project updates by watching https://github.com/bitnami/containers
spark 04:51:39.78 Submit issues and feature requests at https://github.com/bitnami/containers/issues
spark 04:51:39.78

jfang757@cloudshell:~ (cs570jf)$
```

5. Add a test file with a line of words for the word count test
   echo "how much wood could a woodpecker chuck if a woodpecker could chuck wood" > /tmp/test.txt

6. Copy the JAR file containing the application, and any other required files, to the PVC using the mount point
   kubectl cp /tmp/my.jar spark-data-pod:/data/my.jar
   kubectl cp /tmp/test.txt spark-data-pod:/data/test.txt

7. Make sure the files is inside the persistent volume
   kubectl exec -it spark-data-pod -- ls -al /data

```
jfang757@cloudshell:~ (cs570jf)$ echo "how much wood could a woodpecker chuck if a woodpecker could chuck wood" > /tmp/test.txt
jfang757@cloudshell:~ (cs570jf)$ kubectl cp /tmp/my.jar spark-data-pod:/data/my.jar
jfang757@cloudshell:~ (cs570jf)$ kubectl cp /tmp/test.txt spark-data-pod:/data/test.txt
jfang757@cloudshell:~ (cs570jf)$ kubectl exec -it spark-data-pod -- ls -al /data
total 1540
drwxrwsrwx 2 root root    4096 Jul 13 04:54 .
drwxr-xr-x 1 root root    4096 Jul 13 04:49 ..
-rw-r--r-- 1 1001 root 1564259 Jul 13 04:53 my.jar
-rw-r--r-- 1 1000 1000      72 Jul 13 04:54 test.txt
```

8. Deploy ApacheSpark on Kubernetes using the shared volume
   spark-chart.yaml
   vim spark-chat.yaml

Cloud Shell Editor

(cs570jf)

```
service:
  type: LoadBalancer
worker:
  replicaCount: 3
  extraVolumes:
    - name: spark-data
      persistentVolumeClaim:
        claimName: spark-data-pvc
  extraVolumeMounts:
    - name: spark-data
      mountPath: /data
~
~
~
```

9. Deploy Apache Spark on the Kubernetes cluster using the Bitnami Apache Spark Helm chart and supply it with the configuration file above
   helm repo add bitnami https://charts.bitnami.com/bitnami
   helm install spark bitnami/spark -f spark-chart.yaml

```
jfang757@cloudshell:~ (cs570jf)$ helm repo add bitnami https://charts.bitnami.com/bitnami
"bitnami" has been added to your repositories
jfang757@cloudshell:~ (cs570jf)$ helm install spark bitnami/spark -f spark-chart.yaml
NAME: spark
LAST DEPLOYED: Thu Jul 13 05:02:19 2023
NAMESPACE: default
STATUS: deployed
REVISION: 1
TEST SUITE: None
NOTES:
CHART NAME: spark
CHART VERSION: 7.1.0
APP VERSION: 3.4.1

** Please be patient while the chart is being deployed **

1. Get the Spark master WebUI URL by running these commands:

    NOTE: It may take a few minutes for the LoadBalancer IP to be available.
    You can watch the status of by running 'kubectl get --namespace default svc -w spark-master-svc'

  export SERVICE_IP=$(kubectl get --namespace default svc spark-master-svc -o jsonpath="{.status.loadBalancer.ingress[0]['ip', 'hostname'] }")
  echo http://$SERVICE_IP:80

2. Submit an application to the cluster:

  To submit an application to the cluster the spark-submit script must be used. That script can be
  obtained at https://github.com/apache/spark/tree/master/bin. Also you can use kubectl run.

  Run the commands below to obtain the master IP and submit your application.

  export EXAMPLE_JAR=$(kubectl exec -ti --namespace default spark-worker-0 -- find examples/jars/ -name 'spark-example*\.jar' | tr -d '\r')
  export SUBMIT_IP=$(kubectl get --namespace default svc spark-master-svc -o jsonpath="{.status.loadBalancer.ingress[0]['ip', 'hostname'] }")

  kubectl run --namespace default spark-client --rm --tty -i --restart='Never' \
    --image docker.io/bitnami/spark:3.4.1-debian-11-r0 \
    -- spark-submit --master spark://$SUBMIT_IP:7077 \
    --deploy-mode cluster \
    --class org.apache.spark.examples.SparkPi \
    $EXAMPLE_JAR 1000

** IMPORTANT: When submit an application the --master parameter should be set to the service IP, if not, the application will not resolve the master. **
jfang757@cloudshell:~ (cs570jf)$
```

10. Get the external IP of the running pod
kubectl get svc -l "app.kubernetes.io/instance=spark,app.kubernetes.io/name=spark"

```
** IMPORTANT: When submit an application the --master parameter should be set to the service IP, if not, the application will not resolve the master. **
jfang757@cloudshell:~ (c-570jf)$ kubectl get svc -l "app.kubernetes.io/instance=spark,app.kubernetes.io/name=spark"
NAME               TYPE           CLUSTER-IP     EXTERNAL-IP     PORT(S)                      AGE
spark-headless     ClusterIP      None           <none>          <none>                       60s
spark-master-svc   LoadBalancer   10.0.14.133    34.94.73.224    7077:30879/TCP,80:31509/TCP  60s
```

Word Count on Spark

1. Submit the word count task
   Following this example

```
kubectl run --namespace default spark-client --rm --tty -i --restart='Never' \
  --image docker.io/bitnami/spark:3.4.1-debian-11-r0 \
  -- spark-submit --master spark://$SUBMIT_IP:7077 \
  --deploy-mode cluster \
  --class org.apache.spark.examples.SparkPi \
  $EXAMPLE_JAR 1000
```

   My external IP is 34.94.73.224, so $SUBMIT_IP:7077 is 34.94.73.224:7077

   kubectl run --namespace default spark-client --rm --tty -i --restart='Never' \
   --image docker.io/bitnami/spark:3.4.1-debian-11-r0 \
   -- spark-submit --master spark://34.94.73.224:7077 \
   --deploy-mode cluster \
   --class org.apache.spark.examples.JavaWordCount \
   /data/my.jar /data/test.txt

```
jfang757@cloudshell:~ (cs570jf)$ kubectl run --namespace default spark-client --rm --tty -i --restart='Never' --image docker.io/bitnami/spark:3.4.1-debian-11-r0 -- spark-submit --master spark://34.94.73.224:7
77 --deploy-mode cluster --class org.apache.spark.examples.JavaWordCount /data/my.jar /data/test.txt
If you don't see a command prompt, try pressing enter.
23/07/13 06:10:43 INFO SecurityManager: Changing view acls to: spark
23/07/13 06:10:43 INFO SecurityManager: Changing modify acls to: spark
23/07/13 06:10:43 INFO SecurityManager: Changing view acls groups to:
23/07/13 06:10:43 INFO SecurityManager: Changing modify acls groups to:
23/07/13 06:10:43 INFO SecurityManager: SecurityManager: authentication disabled; ui acls disabled; users with view permissions: spark; groups with view permissions: EMPTY; users with modify permissions: spar
k; groups with modify permissions: EMPTY
23/07/13 06:10:43 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
23/07/13 06:10:44 INFO Utils: Successfully started service 'driverClient' on port 41415.
23/07/13 06:10:44 INFO TransportClientFactory: Successfully created connection to /34.94.73.224:7077 after 49 ms (0 ms spent in bootstraps)
23/07/13 06:10:44 INFO ClientEndpoint: ... waiting before polling master for driver state
23/07/13 06:10:44 INFO ClientEndpoint: Driver successfully submitted as driver-20230713061044-0000
23/07/13 06:10:49 INFO ClientEndpoint: State of driver-20230713061044-0000 is RUNNING
23/07/13 06:10:49 INFO ClientEndpoint: Driver running on 10.124.2.6:41407 (worker-20230713060536-10.124.2.6-41407)
23/07/13 06:10:49 INFO ClientEndpoint: spark-submit not configured to wait for completion, exiting spark-submit JVM.
23/07/13 06:10:49 INFO ShutdownHookManager: Shutdown hook called
23/07/13 06:10:49 INFO ShutdownHookManager: Deleting directory /tmp/spark-03b79087-693c-4f69-8d86-9c0e9ad8aebb
pod "spark-client" deleted
jfang757@cloudshell:~ (cs570jf)$
```

Task finished

Use the external IP to open the browser tab

34.94.73.224

Spark 3.4.1

Spark Master at spark://spark-master-0.spark-headless.default.svc.cluster.local:7077

URL: spark://spark-master-0.spark-headless.default.svc.cluster.local:7077
Alive Workers: 3
Cores in use: 6 Total, 0 Used
Memory in use: 43.9 GiB Total, 0.0 B Used
Resources in use:
Applications: 0 Running, 1 Completed
Drivers: 0 Running, 1 Completed
Status: ALIVE

Workers (3)

Worker Id	Address	State	Cores	Memory	Resources
worker-20230713060536-10.124.2.6-41407	10.124.2.6:41407	ALIVE	2 (0 Used)	14.6 GiB (0.0 B Used)	
worker-20230713060644-10.124.0.4-46361	10.124.0.4:46361	ALIVE	2 (0 Used)	14.6 GiB (0.0 B Used)	
worker-20230713060756-10.124.1.9-40765	10.124.1.9:40765	ALIVE	2 (0 Used)	14.6 GiB (0.0 B Used)	

Running Applications (0)

Application ID	Name	Cores	Memory per Executor	Resources Per Executor	Submitted Time	User	State	Duration

Running Drivers (0)

Submission ID	Submitted Time	Worker	State	Cores	Memory	Resources	Main Class	Duration

Completed Applications (1)

Application ID	Name	Cores	Memory per Executor	Resources Per Executor	Submitted Time	User	State	Duration
app-20230713061051-0000	JavaWordCount	5	1024.0 MiB		2023/07/13 06:10:51	spark	FINISHED	20 s

Completed Drivers (1)

Submission ID	Submitted Time	Worker	State	Cores	Memory	Resources	Main Class
driver-20230713061044-0000	2023/07/13 06:10:44	worker-20230713060536-10.124.2.6-41407	FINISHED	1	1024.0 MiB		org.apache.spark.examples.JavaWordCount

2. Get the name of the worker node
   kubectl get pods -o wide | grep 10.124.2.6

```
jfang757@cloudshell:~ (cs570jf)$ kubectl get pods -o wide | grep 10.124.2.6
spark-worker-0                1/1     Running   0          9m34s   10.124.2.6   gke-w7hl-default-pool-6b3f8182-l81r   <none>           <none>
jfang757@cloudshell:~ (cs570jf)$ kubectl exec -it spark-worker-0 -- bash
I have no name!@spark-worker-0:/opt/bitnami/spark$ cd /opt/bitnami/spark/work
```

3. Execute this pod and check the result of the finished tasks
   kubectl exec -it spark-worker-0 -- bash
   cd /opt/bitnami/spark/work
   cat driver-20230713061044-0000/stdout

```
I have no name!@spark-worker-0:/opt/bitnami/spark/work$ cat driver-20230713061044-0000/stdout
if: 1
a: 2
how: 1
could: 2
wood: 2
woodpecker: 2
much: 1
chuck: 2
I have no name!@spark-worker-0:/opt/bitnami/spark/work$
```

4. Exit the current session
   exit

```
I have no name!@spark-worker-0:/opt/bitnami/spark/work$ exit
exit
command terminated with exit code 127
jfang757@cloudshell:~ (cs570jf)$
```

Running python PageRank on PySpark on the pods

1. Running python PageRank on PySpark on the pods
   Back to spark master pods
   kubectl exec -it spark-master-0 -- bash

   Go to the directory where pagerank.py located
   cd /opt/bitnami/spark/examples/src/main/python

   Run the pagerank
   spark-submit pagerank.py /opt 2