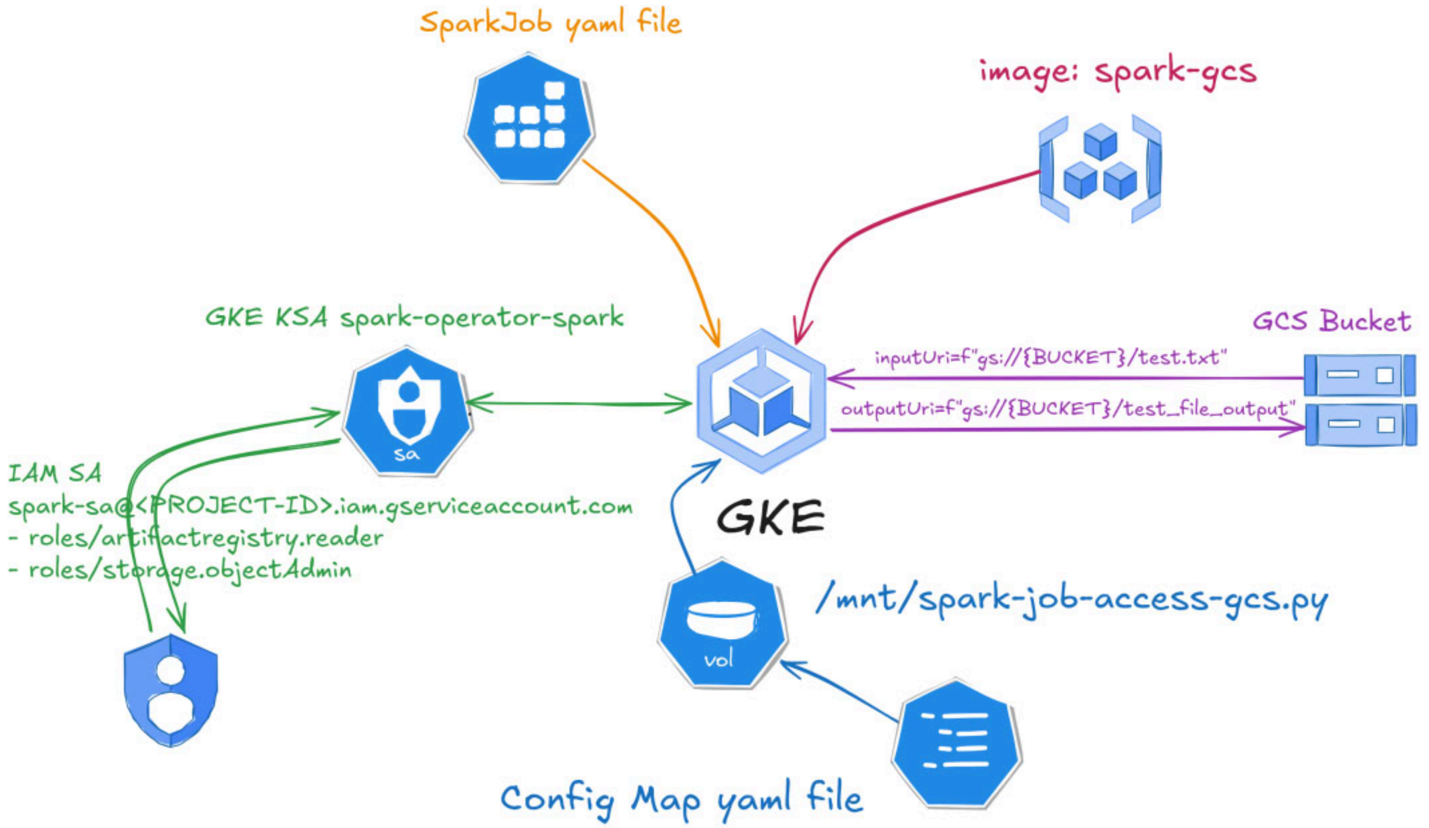
Spark on Google GKE

In this Demo SparkJob, a script is implemented that accesses a Google Bucket, reads a file, and writes the processing result back to the Google Bucket.


SparkJob yaml file
image: spark-gcs
GKE KSA spark-operator-spark
sa
GCS Bucket
inputUri=f"gs://{BUCKET}/test.txt"
outputUri=f"gs://{BUCKET}/test_file_output"
IAM SA
spark-sa@<PROJECT-ID>.iam.gserviceaccount.com
- roles/artifactregistry.reader
- roles/storage.objectAdmin
GKE
vol
/mnt/spark-job-access-gcs.py
Config Map yaml file

```
data:
  spark-job-access-gcs.py: |
    import pyspark
    import sys
    BUCKET = "<NAME>"
    inputUri=f"gs://{BUCKET}/test.txt"
    outputUri=f"gs://{BUCKET}/test_file_output"
    sc = pyspark.SparkContext()
    lines = sc.textFile(inputUri)
    words = lines.flatMap(lambda line: line.split())
    wordCounts = words.map(lambda word: (word, 1)).reduceByKey(lambda count1, count2: count1 + count2)
    wordCounts.saveAsTextFile(outputUri)
```

GKE Installation

Create a basic GKE cluster autopilot-cluster-1 with my favorite autopilot mode

```
gcloud beta container --project "<PROJECT_ID>" \
clusters create-auto "autopilot-cluster-1" \
--region "us-central1" \
--release-channel "regular" \
--tier "standard" \
--enable-ip-access --no-enable-google-cloud-access \
--network "projects/<PROJECT_ID>/global/networks/default" \
--subnetwork "projects/<PROJECT_ID>/regions/us-central1/subnetworks/default" \
--cluster-ipv4-cidr "/17" --binauthz-evaluation-mode=DISABLED
```

```
# Connecting to the cluster
gcloud container clusters get-credentials autopilot-cluster-1 --region us-central1 --project <PROJECT_ID>
```

Spark Application Helm Chart installation

Performing a basic installation of Spark Application

```
# Let's add the spark-operator repository
helm repo add spark-operator https://kubeflow.github.io/spark-operator
helm repo update
```

```
# Install Spark Application in namespace default
helm install spark-operator spark-operator/spark-operator --namespace spark-operator --create-namespace --set webhook.enable=true
```

I would like to draw special attention to the webhook parameter. It is Webhook that allows you to mount volumes (pvc, config-map, GCS) in SparkJob in driver executor pods.

Check the installation. If the installation is successful, two deployments should be present and running successfully in the default namespace.

```
kubectl get deployments -n default
NAME                         READY   UP-TO-DATE   AVAILABLE   AGE
spark-operator-controller    1/1     1            1           60s
spark-operator-webhook       1/1     1            1           60s

# Checking for the presence of Object Kind: SparkApplication
kubectl explain Sparkapplication -n default
GROUP:      sparkoperator.k8s.io
KIND:       SparkApplication
VERSION:    v1beta2
```

Workload Identity. Link KSA with IAM SA

```
# After installation several Kubernetes Service Accounts (KSAs) must also be created.
kubectl get sa -n default
NAME                         SECRETS   AGE
default                      0         7m30s
spark-operator-controller    0         18s
spark-operator-spark         0         18s
spark-operator-webhook       0         18s
```

KSA **spark-operation-spark** that will be used in our SparkJob

Preparation Spark image to work with GCP Services

Spark basic image by default does not know how to work with Google Cloud Storage gs-filesystem and does not know how to work with Big Query. To work with these services, you need to build a custom docker image with the necessary libraries using this Dockerfile

```
# Build docker image
git pull https://github.com/dnk80/GKE-spark.git && cd GKE-spark/
docker build . -t us-central1-docker.pkg.dev/<PROJECT_ID>/spark/spark-gcs:3.5.3

# Push builded docker image in Google Artifactory (Artifactory spark need to created before)
docker push us-central1-docker.pkg.dev/<PROJECT_ID>/spark/spark-gcs:3.5.3
```

To successfully launch a SparkJob with a Custom Image using the spark-operation-spark service account, we need to use Workload Identity and RBAC to link Kubernetes Service Accounts to the IAM Service Account by providing the following set of roles:

- roles/artifactregistry.reader # To use a custom image in SparkJob
- roles/storage.objectAdmin # For read/write access to data in GCS

Create a service account and assign on it the necessary roles

```
gcloud projects add-iam-policy-binding <PROJECT_ID> \
--member='serviceAccount:spark-sa@<PROJECT_ID>.iam.gserviceaccount.com' \
--role='roles/artifactregistry.reader'

gcloud projects add-iam-policy-binding <PROJECT_ID> \
--member='serviceAccount:spark-sa@<PROJECT_ID>.iam.gserviceaccount.com' \
--role='roles/storage.objectAdmin'
```

Linking KSA spark-operation-spark to GCP IAM SA

```
gcloud iam service-accounts add-iam-policy-binding \
--role roles/iam.workloadIdentityUser \
--member "serviceAccount:<PROJECT_ID>.svc.id.goog[default/spark-operator-spark]" \
spark-sa@<PROJECT_ID>.iam.gserviceaccount.com
```

And don't forget about the annotation

```
kubectl annotate serviceaccount spark-operator-spark \
iam.gke.io/gcp-service-account=spark-sa@<PROJECT_ID>.iam.gserviceaccount.com \
-n default
```

Run SparkJob

Almost everything is ready. Let's install all the necessary components to run SparkJob. Let's create a GCS bucket. Created test.txt file with data and copied it into GCS bucket test-gcs-418919. If you plan use another name of bucket or filename don`t forgot change script in config-map.yaml on proper names

```
gsutil mb gs://test-gcs-418919/
echo "cat dog elephant fish" > test.txt
gsutil cp test.txt gs://test-gcs-418919/
```

Create config-map with demo script. Данная config-map будет преобразована в python файл и подключена к контейнеру как том папка с файлом /mnt/spark-job-access-gcs.py. Данный файл является точкой входа для запуска Spark Job.

```
kubectl apply -f config-map.yaml
configmap/py-script-map created
```

Run Spark Job

```
kubectl apply -f spark.yaml
sparkapplication.sparkoperator.k8s.io/spark-test created
```

Let's check how Spark Job executed

```
kubectl describe sparkapplication spark-test
Events:
  Type    Reason                       Age    From                            Message
  ----    ------                       ----   ----                            -------
  Normal  SparkApplicationSubmitted    5m33s  spark-application-controller  SparkApplication spark-test was submitted successfully
  Normal  SparkDriverRunning           3m58s  spark-application-controller  Driver spark-test-driver is running
  Normal  SparkExecutorPending         2m57s  spark-application-controller  Executor [spark-test-be7d2e950c958d81-exec-1] is pending
  Normal  SparkExecutorRunning         98s    spark-application-controller  Executor [spark-test-be7d2e950c958d81-exec-1] is running
  Normal  SparkDriverCompleted         37s    spark-application-controller  Driver spark-test-driver completed
  Normal  SparkExecutorCompleted       36s    spark-application-controller  Executor [spark-test-be7d2e950c958d81-exec-1] completed
```

```
Check artifacts in the GCS bucket after SparkJob finished.

gsutil ls gs://test-gcs-418919/test_file_output/
gs://test-gcs-418919/test_file_output/
gs://test-gcs-418919/test_file_output/_SUCCESS
gs://test-gcs-418919/test_file_output/part-00000
gs://test-gcs-418919/test_file_output/part-00001
```