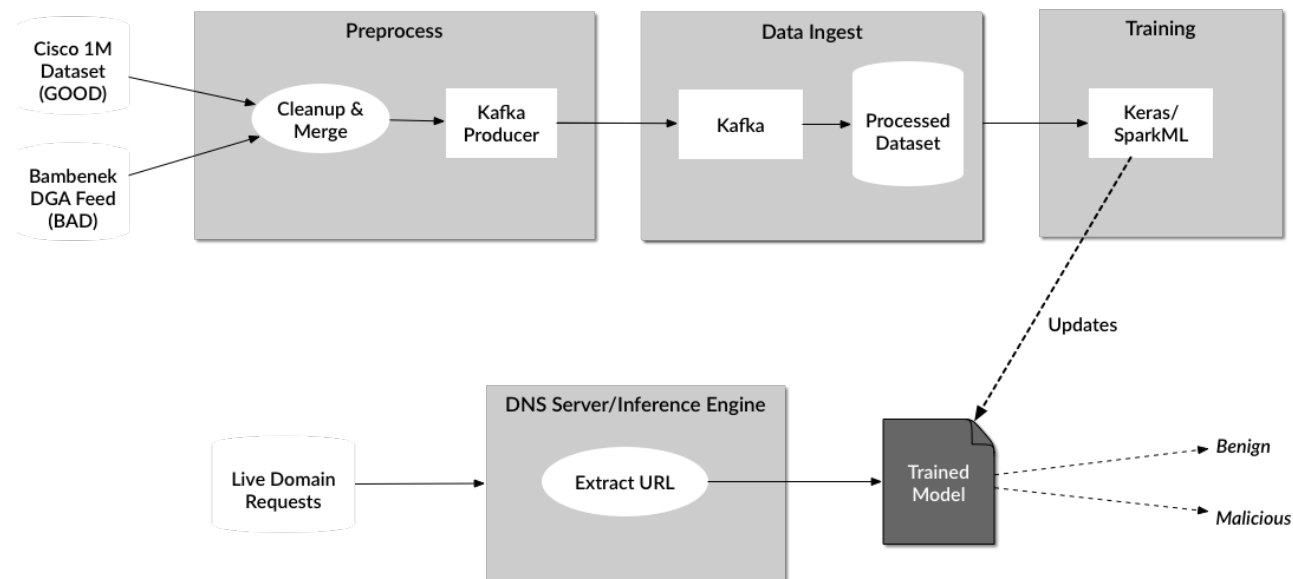


Setting up Environment in GCP

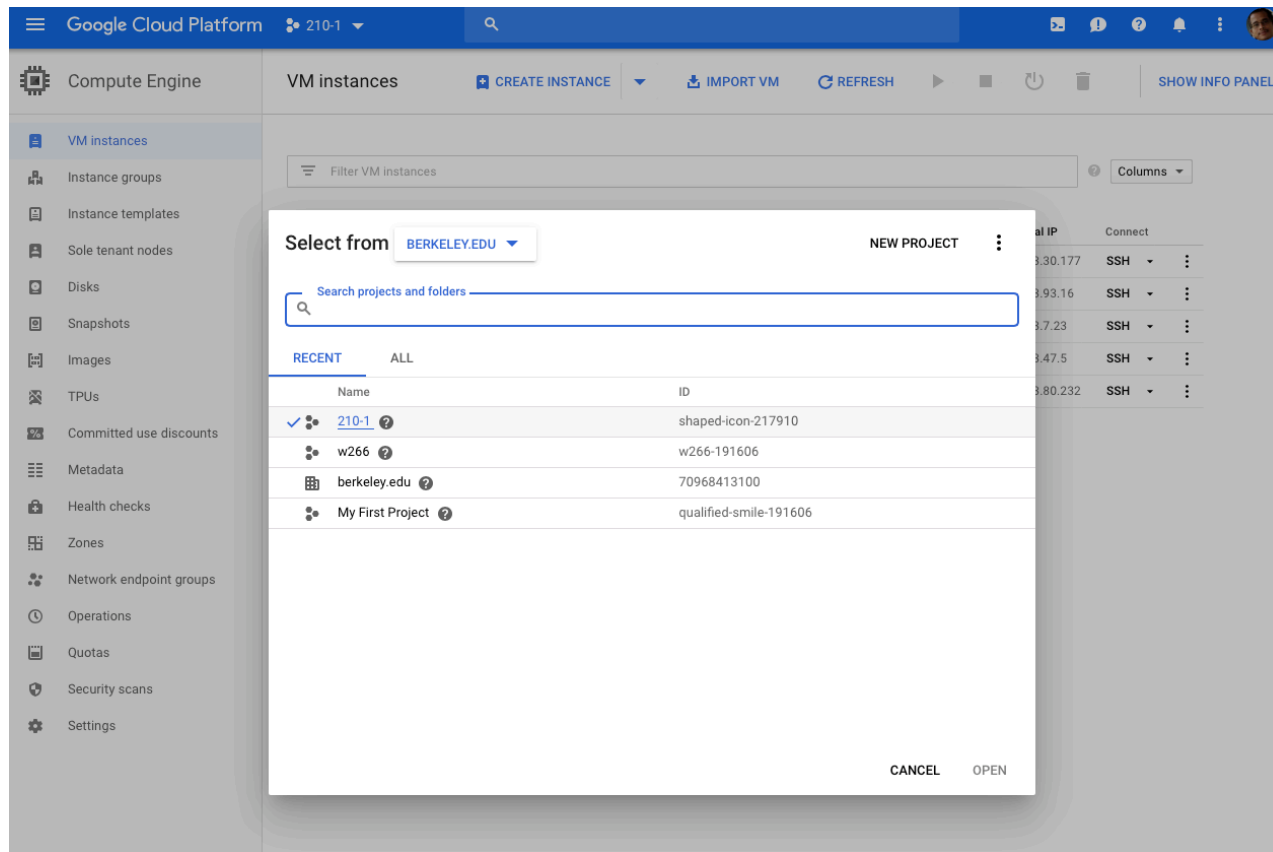
End Goal

To setup a distributed scalable environment on GCP for DGA Feed Training and Inference Engine that looks like this.



Prerequisites

- Setup GCP account
- Installed Google Cloud SDK on local machine (<https://cloud.google.com/sdk/install>)
- Created a project



Kubernetes setup

- On your browser go to kubernetes engine page and select the project (This will take several mins to start kubernetes engine)
https://console.cloud.google.com/projectselector/kubernetes?_ga=2.201928657.1707404544.1516051830

- While kubernetes engine gets ready, on your local machine, pull the docker images

```
$ docker pull confluentinc/cp-zookeeper:latest
$ docker pull confluentinc/cp-kafka:latest
$ docker pull midsw205/cdh-minimal:latest
$ docker pull midsw205/spark-python:0.0.5
$ docker pull midsw205/base:0.1.9
```

- Check the images

```
$ docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
confluentinc/cp-kafka	latest	373a4e31e02e	6 weeks ago	558MB
confluentinc/cp-zookeeper	latest	18b57832a1e2	4 months ago	562MB
midsw205/spark-python	latest	a554c3520502	6 months ago	3.87GB
midsw205/base	latest	03fff049d97a	6 months ago	2.7GB

- Install kubectl

```
$ gcloud components install kubectl
```

- Tag the images to be pushed to GCP by using the image ids seen in the docker images output above.

```
$ docker tag a554c3520502 gcr.io/w210-1/spark-python
$ docker tag 03fff049d97a gcr.io/w210-1/mids
$ docker tag 373a4e31e02e gcr.io/w210-1/kafka
$ docker tag 18b57832a1e2 gcr.io/w210-1/zookeeper
```

- Push the tagged images to gcloud

```
$ gcloud docker --push gcr.io/w210-1/spark-python
$ gcloud docker --push gcr.io/w210-1/mids
$ gcloud docker --push gcr.io/w210-1/kafka
$ gcloud docker --push gcr.io/w210-1/zookeeper
```

- Create a cluster in gcloud

```
$ gcloud container clusters create kafka --num-nodes=5 --zone northamerica-northeast1-a
```

This creates a cluster with 5 nodes below (5 nodes is the maximum in n1-standard-1 flavor). You can change the zone based on where you live.

- Check if the cluster is created and the computes are operational:

```
$ gcloud compute instances list
```

NAME	ZONE	MACHINE_TYPE	PREEMPTIBLE	INTERNAL_IP
EXTERNAL_IP STATUS				
gke-kafka-default-pool-6a5787d6-299m 35.203.30.177 RUNNING	northamerica-northeast1-a	n1-standard-1		10.162.0.6
gke-kafka-default-pool-6a5787d6-4g14 35.203.93.16 RUNNING	northamerica-northeast1-a	n1-standard-1		10.162.0.2
gke-kafka-default-pool-6a5787d6-l11c 35.203.7.23 RUNNING	northamerica-northeast1-a	n1-standard-1		10.162.0.4
gke-kafka-default-pool-6a5787d6-pwq6 35.203.47.5 RUNNING	northamerica-northeast1-a	n1-standard-1		10.162.0.5
gke-kafka-default-pool-6a5787d6-vczj 35.203.80.232 RUNNING	northamerica-northeast1-a	n1-standard-1		10.162.0.3

Deploying the containers

- We will use the following files in gcp directory

```
$ ls -l *.yaml
kafka-deployment.yaml
kafka-service.yaml
mids-claim0-persistentvolumeclaim.yaml
mids-deployment.yaml
mids-service.yaml
myhdfs-deployment.yaml
myhdfs-service.yaml
myspark-deployment.yaml
myspark-service.yaml
zookeeper-deployment.yaml
zookeeper-service.yaml
```

- Use kubectl to bringup service and deployments

```
kubectl create --filename zookeeper-deployment.yaml
kubectl create --filename kafka-deployment.yaml
kubectl create --filename mids-deployment.yaml
kubectl create --filename myspark-deployment.yaml
kubectl create --filename zookeeper-service.yaml
kubectl create --filename kafka-service.yaml
kubectl create --filename mids-service.yaml
kubectl create --filename myspark-service.yaml
```

- Check if all the containers are deployed

```
$ kubectl get pods -o wide
```

NAME	READY	STATUS	RESTARTS	AGE	IP	NODE
kafka-5c7bb56cbd-8ndmk4gl4	1/1	Running	0	4d	10.20.0.4	gke-kafka-default-pool-6a5787d6-
mids-545f9676c-z2jvf111c	1/1	Running	0	4d	10.20.2.5	gke-kafka-default-pool-6a5787d6-
myspark-cd94d8765-vvdf8pwq6	1/1	Running	0	2m	10.20.1.7	gke-kafka-default-pool-6a5787d6-
zookeeper-6cbcd499f-wmsnqpwq6	1/1	Running	0	5d	10.20.1.5	gke-kafka-default-pool-6a5787d6-

Verifying the message flow

- Coming soon..