Warsaw University of Technology
Faculty of Mathematics and Information Science

Lab 6

Compute Resources - Spark/Kubernetes

dr Elena Konecka, PhD
Elena.Konetskaia@pw.edu.pl

Compute Resources

Spark/Kubernetes

1. Installing Kubernetes, Spark
2. Configuration files, project structure
3. Simplest Spark app in Kubernetes (single-node mode)
4. Spark app in Kubernetes (multi-node mode)

Installing Kubernetes, Spark

Elena Konecka

kind and kubectl

```
curl -Lo ./kind https://kind.sigs.k8s.io/dl/v0.25.0/kind-linux-amd64

chmod +x ./kind

sudo mv ./kind /usr/local/bin/kind

kind --version
```

Main commands for kind:

```
kind get clusters

kind get nodes --name <cluster_name>

kind get kubeconfig --name <cluster_name>

kind delete cluster --name <cluster_name>
```

```
sudo snap install kubectl --classic

kubectl version --client
```

Elena Konecka

Spark and spark-submit

```
wget
https://archive.apache.org/dist/spark/spark-3.5.0/spark-3.5.0-bin-hadoop3.tgz
```

```
tar -xvzf spark-3.5.0-bin-hadoop3.tgz
```

```
mv spark-3.5.0-bin-hadoop3 spark
```

Spark contains Dockerfile, co let’s move to the folder and build the image

```
cd spark
```

```
./bin/docker-image-tool.sh -r spark -t v3.5.0 build
```

Remember, here everywhere we use Spark v3.5.0!

Configuration files, project structure

Elena Konecka

Project structure

```
<last_name>__cluster/
└─ cluster/
└─ manifests/
└─ spark/
    └─ app/
    └─ input/
```

Elena Konecka

Users, manifests, etc

Simplest Spark app in Kubernetes (single-node mode)

Elena Konecka

Create a config file for cluster with one node

cluster/kind-cluster.yaml

```yaml
kind: Cluster
apiVersion: kind.x-k8s.io/v1alpha4
nodes:
  - role: control-plane
    extraPortMappings:
      - containerPort: 30080
        hostPort: 8080
        protocol: TCP
```

Create a cluster being in project root:

```
kind create cluster --name spark-cluster --config cluster/kind-cluster.yaml
```

```
kubectl get nodes
```

You have to find one node with status READY

Then create namespace:

```
kubectl create namespace spark
```

Elena Konecka

Create the simple spark app and input data file

spark/app/simple_csv_app.py

```
from pyspark.sql import SparkSession
from pyspark.sql.functions import avg, col
import sys

def main():
    input_path = sys.argv[1] if len(sys.argv) > 1 else "/input/data.csv"

    spark =
SparkSession.builder.appName("SimpleCSVProcessing").getOrCreate()
    df = spark.read.option("header", "true").option("inferSchema",
"true").csv(input_path)
    print("=== Original data ===")
    df.show()
    avg_score = df.agg(avg("score")).collect()[0][0]
    print(f"Average score = {avg_score}")
    above = df.filter(col("score") > avg_score)
    print("=== Higher then average: ===")
    above.show()

    spark.stop()

if __name__ == "__main__":
    main()
```

Rename in this code the your app (SimpleCSVProcessing -> <last_name>_simple_app). Check if you should to change anything in the code of spark-submit

spark/input/data.csv

```
name,age,score
Alice,20,85
Bob,21,90
Charlie,19,78
Diana,22,92
Eve,20,88
```

And then you can check if app works:

```
spark-submit spark/app/simple_csv_app.py
spark/input/data.csv
```

Please read logs which appear at the moment when app starts

Elena Konecka

Preparing a Docker image

spark/Dockerfile

```
FROM apache/spark:3.5.0

COPY app/simple_csv_app.py /opt/spark/app.py

COPY input/data.csv /data/data.csv
```

Build the image:

```
cd spark

docker build -t spark-local:latest .
```

Check if it had been built:

```
docker images | grep spark-local
```

Spread the image among the nodes:

```
kind load docker-image spark-local:latest --name spark-cluster
```

Now you can run the task in the Kubernetes cluster executing command in terminal:

```
spark-submit \
  --master k8s://https://$(kubectl config view --minify -o jsonpath='{.clusters[0].cluster.server}' | sed 's|https://||') \
  --deploy-mode cluster \
  --name simple-csv-app \
  --class org.apache.spark.deploy.PythonRunner \
  --conf spark.executor.instances=2 \
  --conf spark.kubernetes.container.image=spark-local:latest \
--conf spark.kubernetes.container.image.pullPolicy=Never \
  local:///opt/spark/app.py /data/data.csv
```

or creating an executable bash script

Assignment 4: On this step appears a problem related to the Docker image. Please find a problem and solve it using this resource: https://kind.sigs.k8s.io/docs/user/quick-start

Take into account that the same error presents in all spark-submit! Be careful!

Elena Konecka

Getting result locally

In simple_csv_app.py replace the last block: above.write.mode("overwrite").csv("/output/result")

And add the volume:

```
spark-submit \

  --master k8s://$(kubectl config view ... ) \

  --deploy-mode cluster \

  --conf spark.kubernetes.container.image=spark-local:latest \

  --conf spark.kubernetes.driver.volumes.hostPath.output.mount.path=/output \

  --conf
spark.kubernetes.driver.volumes.hostPath.output.options.path=/tmp/spark-output \

  --conf spark.kubernetes.executor.volumes.hostPath.output.mount.path=/output \

  --conf
spark.kubernetes.executor.volumes.hostPath.output.options.path=/tmp/spark-output \

  local:///opt/spark/app.py /data/data.csv
```

Locally you will find the output file:

```
ls /tmp/spark-output/result
```

Elena Konecka

Updating Docker Image

Please add some extra calculations into the spark/app/simple_csv_app.py.

Rebuild the image: docker build -t spark-local:latest .

kind load docker-image spark-local:latest --name spark-cluster

and then execute the same **spark-submit**

Spark app in Kubernetes (multi-node mode)

Elena Konecka

Create a config file for cluster with one node

cluster/kind-cluster-multinode.yaml

```yaml
kind: Cluster
apiVersion: kind.x-k8s.io/v1alpha4
nodes:
  - role: control-plane
    extraPortMappings:
      - containerPort: 30080
        hostPort: 8080
        protocol: TCP
  - role: worker
  - role: worker
```

Delete old cluster and create a new one being in project root:

```
kind delete cluster --name spark-cluster
```

```
kind create cluster --name spark-cluster
--config cluster/kind-cluster-multinode.yaml
```

```
kubectl get nodes
```

You have to find one node with status READY

Then create namespace:

```
kubectl create namespace spark
```

Elena Konecka

Checking Pod Distribution

1. Create a test deployment to see distribution:

```
kubectl create deployment nginx
--image=nginx
```

```
kubectl scale deployment nginx --replicas=5
```

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

The NODE column shows which node each pod is running on

This demonstrates load distribution across nodes

2. Prepare the Spark Docker image similarly to the slide 12:

```
cd spark
```

```
docker build -t spark-local:latest .
```

```
kind load docker-image spark-local:latest
--name spark-cluster
```

In multi-node Kind, all nodes can use local images; Kind automatically makes the image available on worker nodes.

Running Spark Application with Multiple Executors

```
spark-submit \

  --master k8s://https://$(kubectl config view --minify -o
jsonpath='{.clusters[0].cluster.server}' | sed 's|https://||') \

  --deploy-mode cluster \

  --name simple-csv-app \

  --class org.apache.spark.deploy.PythonRunner \

  --conf spark.executor.instances=4 \

  --conf spark.kubernetes.container.image=spark-local:latest \

  local:///opt/spark/app.py /data/data.csv
```

We request 4 executors, which will be distributed across worker nodes

Check distribution:

```
kubectl get pods -n spark -o wide
```

Elena Konecka

Practical task

- Calculate the average and maximum of the score column
- Find students with scores above the average
- Check which nodes executors are running on
- Change a cluster creating one control-plane node and one worker node , recreate the cluster, and observe changes in distribution
- Update the Docker image and verify that all executors run the updated code

Elena Konecka

Turning off the VM

Don’t forget to close ssh connection:

exit

and turn off the VM on the OS level type (in the VM Box window):

shutdown –h 0