Lesson 04 Demo 04

Creating a Multi-Container Pod

Objective: To create a multi-container pod in a Kubernetes cluster, allowing you to run and interact with multiple containers within a single pod

Tools required: kubeadm, kubectl, kubelet, and containerd

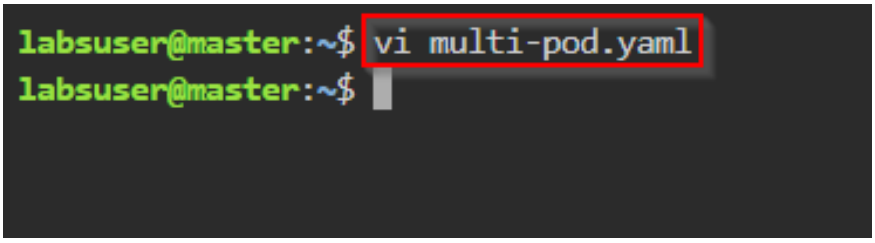
Prerequisites: A Kubernetes cluster should already be set up (refer to the steps in Lesson 02, Demo 01 for guidance).

Steps to be followed:

1. Create and access a multi-container pod

Step 1: Create and access a multi-container pod

1.1 Create a YAML file by entering the following command:
vi multi-pod.yaml



1.2 Copy and paste the code below into the **multi-pod.yaml,** and then save it:

```
apiVersion: v1
kind: Pod
metadata:
  name: multicontainer-pod
spec:
  containers:
  #Container 01
  - name: web
  image: nginx
```

```
  ports:
  - containerPort: 80
#Container 02
- name: redis
  image: redis
  ports:
  - containerPort: 6379
```

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: multicontainer-pod
spec:
  containers:
  #Container 01
  - name: web
    image: nginx
    ports:
    - containerPort: 80
  #Container 02
  - name: redis
    image: redis
    ports:
    - containerPort: 6379
```

1.3 Create the pod by entering the following command:
kubectl apply -f multi-pod.yaml

```
labsuser@master:~$ vi multi-pod.yaml
labsuser@master:~$ kubectl apply -f multi-pod.yaml
pod/multicontainer-pod created
labsuser@master:~$
```

1.4 Execute the command inside the **web** container of the **multicontainer-pod** to display the date and time:
kubectl exec -it multicontainer-pod -c web -- date

```
labsuser@master:~$ vi multi-pod.yaml
labsuser@master:~$ kubectl apply -f multi-pod.yaml
pod/multicontainer-pod created
labsuser@master:~$ kubectl exec -it multicontainer-pod -c web -- date
Mon Oct 30 15:43:33 UTC 2023
labsuser@master:~$
```

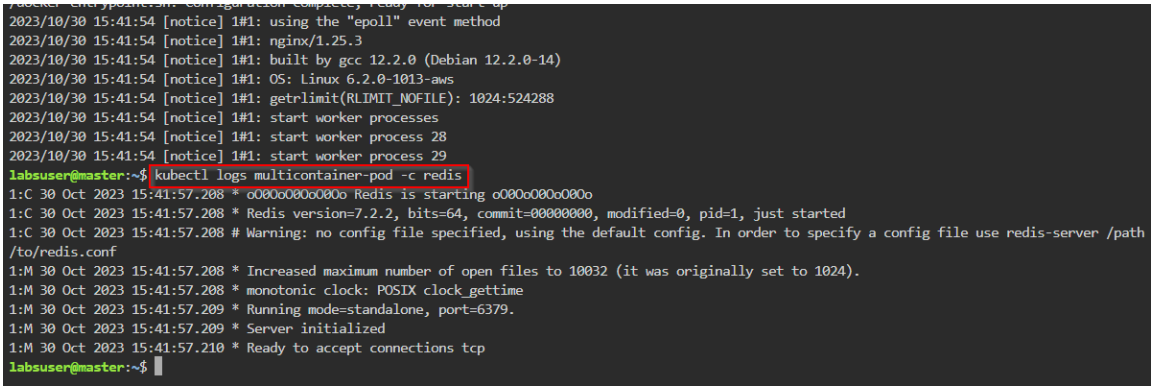
1.5 Execute the command inside the **redis** container of the **multicontainer-pod** to display the date and time:
kubectl exec -it multicontainer-pod -c redis -- date

```
labsuser@master:~$ vi multi-pod.yaml
labsuser@master:~$ kubectl apply -f multi-pod.yaml
pod/multicontainer-pod created
labsuser@master:~$ kubectl exec -it multicontainer-pod -c web -- date
Mon Oct 30 15:43:33 UTC 2023
labsuser@master:~$ kubectl exec -it multicontainer-pod -c redis -- date
Mon Oct 30 15:46:11 UTC 2023
labsuser@master:~$
```

1.6 To view the logs of the **web** container, use the following command:
kubectl logs multicontainer-pod -c web

```
labsuser@master:~$ kubectl exec -it multicontainer-pod -c redis -- date
Mon Oct 30 15:46:11 UTC 2023
labsuser@master:~$ kubectl logs multicontainer-pod -c web
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
/docker-entrypoint.sh: Sourcing /docker-entrypoint.d/15-local-resolvers.envsh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
/docker-entrypoint.sh: Configuration complete; ready for start up
2023/10/30 15:41:54 [notice] 1#1: using the "epoll" event method
2023/10/30 15:41:54 [notice] 1#1: nginx/1.25.3
2023/10/30 15:41:54 [notice] 1#1: built by gcc 12.2.0 (Debian 12.2.0-14)
2023/10/30 15:41:54 [notice] 1#1: OS: Linux 6.2.0-1013-aws
2023/10/30 15:41:54 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1024:524288
2023/10/30 15:41:54 [notice] 1#1: start worker processes
2023/10/30 15:41:54 [notice] 1#1: start worker process 28
2023/10/30 15:41:54 [notice] 1#1: start worker process 29
labsuser@master:~$
```

1.7 View the logs of the **redis** container with the following command:
kubectl logs multicontainer-pod -c redis



Note: When working with a pod that contains only one container, the **-c** flag is not necessary. However, for multi-container pods, specifying the container is crucial.

By following these steps, you have successfully established a multi-container pod within a Kubernetes cluster.