# Storage

kubectl explain pod.spec.volumes

## Non-Persistent/Ephemeral Volume

### emptyDir

apiVersion: v1

kind: Pod

metadata:

name: sharevol

spec:

containers:

- name: c1

image: centos:7

command:

- "bin/bash"

- "-c"

- "sleep 10000"

volumeMounts:

- name: xchange

mountPath: "/tmp/xchange"

- name: c2

image: centos:7

command:

- "bin/bash"

- "-c"

- "sleep 10000"

volumeMounts:

- name: xchange

mountPath: "/tmp/data"

volumes:

- name: xchange

emptyDir: {}

k apply -f pod-vol-emptyDir.yaml

k get pod sharevol

k exec sharevol -c c1 -- ls /tmp

k exec sharevol -c c2 -- ls /tmp

k exec sharevol -c c2 -- touch /tmp/file1

#Delete and recreate the pod to check data

k exec sharevol -c c2 -- ls /tmp

#Data will not be there

### Host Directory

apiVersion: v1

kind: Pod

metadata:

name: webapp

spec:

containers:

- name: event-simulator

image: nginx

volumeMounts:

- name: log-volume

mountPath: /log

volumes:

- name: log-volume

hostPath:

# directory location on host

path: /var/log

# this field is optional

type: Directory

k apply -f pod-vol-dir.yaml

k get pod webapp

k exec webapp -- ls /log

k exec webapp -- touch /log/k8s.txt

k get pods -o wide | grep webapp

#With the worker node identified, Get into that worker node

ls /var/log/

#check for k8s.txt file and add some text

sudo vi /var/log/k8s.txt

k exec webapp -- cat /log/k8s.txt

#file content will be same

## Volume Config

### ConfigMap

apiVersion: v1

kind: Pod

metadata:

name: configmap-pod

spec:

containers:

- name: mypod

image: redis

ports:

- containerPort: 80

volumeMounts :

- name: sample

mountPath: /home/labsuser/configfiles/

readOnly: true

volumes :

- name: sample

configMap:

name: game-demo

k apply -f cm-pod

k exec -it configmap-pod /bin/sh

### Secret

apiVersion: v1

kind: Pod

metadata:

name: secret-pod

spec:

containers:

- name: mypod

image: redis

volumeMounts:

- name: foo

mountPath: "/etc/foo"

readOnly: true

volumes:

- name: foo

secret:

secretName: mysecret

kubectl apply -f sec-pod

k exec -it secret-pod /bin/sh

## Persistent Volume

k explain pod.spec.volumes | grep persistentVolumeClaim

k explain pv.spec

pv.yaml

apiVersion: v1

kind: PersistentVolume

metadata:

name: task-pv-volume

labels:

type: local

spec:

storageClassName: manual

capacity:

storage: 10Gi

accessModes:

- ReadWriteOnce

hostPath:

path: "/mnt/data"

#RWO can be used one containers

k apply -f pv.yaml

k get pv

## Persistent Volume Claim

#PV Status before PVC creation

k get pv

pvc.yaml

apiVersion: v1

kind: PersistentVolumeClaim

metadata:

name: task-pv-claim

spec:

storageClassName: manual

accessModes:

- ReadWriteOnce

resources:

requests:

storage: 3Gi

k apply -f pvc.yaml

#PVC and PV Status after PVC creation

k get pvc

k get pv

#Pod using PVC

#PVC Status before pod creation

k get pvc

pod-pvc.yaml

apiVersion: v1

kind: Pod

metadata:

name: task-pv-pod

spec:

volumes:

- name: task-pv-storage

persistentVolumeClaim:

claimName: task-pv-claim

containers:

- name: task-pv-container

image: nginx

ports:

- containerPort: 80

name: "http-server"

volumeMounts:

- mountPath: "/usr/share/nginx/html"

name: task-pv-storage

k apply -f pod-pvc.yaml

#PVC Status after pod creation

k get pvc

#Verify Volume by creating a file

k exec -it task-pv-pod -- touch /usr/share/nginx/html/index.html

k exec -it task-pv-pod -- ls /usr/share/nginx/html/

Identify the worker node, where pod is running/scheduled

k get pod task-pv-pod -o wide

In worker nodes, Example: worker02

ls /mnt/data

index.html

k delete -f pod-pvc.yaml

Verify file exists in worker node, Example: worker02

ls /mnt/data

index.html

k apply -f pod-pvc.yaml

k exec -it task-pv-pod -- ls /usr/share/nginx/html/

index.html

## AKS - Creating PVC with dynamic PV

Creating sc

apiVersion: storage.k8s.io/v1

kind: StorageClass

metadata:

labels:

addonmanager.kubernetes.io/mode: EnsureExists

kubernetes.io/cluster-service: "true"

name: managed-premium-01

resourceVersion: "391"

uid: 08be1a5c-7fcf-4603-91eb-354e2fe27fa3

parameters:

cachingmode: ReadOnly

kind: Managed

storageaccounttype: Premium\_LRS

provisioner: disk.csi.azure.com

reclaimPolicy: Delete

volumeBindingMode: Immediate

Creating PVC

apiVersion: v1

kind: PersistentVolumeClaim

metadata:

name: azure-managed-disk

spec:

accessModes:

- ReadWriteOnce

storageClassName: managed-premium-01

resources:

requests:

storage: 5Gi

k get pvc

k get pv