Lab - Persistent Volume and LivenessProbe in Kubernetes

09:0

Volumes

- By default the container filesystem is ephemeral – recreated each time when the container starts → a clean state each time → can be a problem for non trivial applications
- A pod can have multiple containers that are sharing files.
- A volume in the simplest form is just a directory which is accessible to the containers in a pod.
- The type of volume determines the backend for the directory.
- The pod definition specifies what volumes are provided (the *spec.volumes* field), and where are these mounted in the containers (the *spec.containers.volumeMounts* field).
- The containers are independently specifying where to mount each volume (the same volume can be mounted on different path in different containers).

Volume example

```
apiVersion: v1
kind: Pod
metadata:
  name: test-pd
spec:
  containers:
  - image: gcr.io/google_containers/test-webserver
    name: test-container
    volumeMounts:
    - mountPath: /cache
      name: cache-volume
  volumes:
  - name: cache-volume
    emptyDir: {}
```

- Pod has volume or attached to pod and shared between pod
- If pod failed - volume associate will get removed
- Only access inside pod
- Shared volume - in 2 containers
-

==========
volume labs
==========

```
apiVersion: v1
kind: Pod
metadata:
  name: myvolemptydir
spec:
  containers:
  - name: c1
    image: centos
    command: ["/bin/bash", "-c", "sleep 15000"]
    volumeMounts:                    # Mount definition inside the container
      - name: xchange
        mountPath: "/tmp/xchange"
```

```
  - name: c2
    image: centos
    command: ["/bin/bash", "-c", "sleep 10000"]
    volumeMounts:
    - name: xchange
      mountPath: "/tmp/data"
  volumes:
  - name: xchange
    emptyDir: {}
```

==========
HOST PATH
==========

```
apiVersion: v1
kind: Pod
metadata:
  name: myvolhostpath
spec:
  containers:
  - image: centos
    name: testc
    command: ["/bin/bash", "-c", "sleep 15000"]
    volumeMounts:
    - mountPath: /tmp/hostpath
      name: testvolume
  volumes:
  - name: testvolume
    hostPath:
      path: /tmp/data
```

Volume types

- Kubernetes supports several volume types:
  - emptyDir – initially empty; deleted when the pod is deleted (survives crashes)
  - hostPath – mounts a directory from the host into the pod. The content is host specific → pods with identical specs can behave differently on different nodes.
  - gcePersistentDisk – mounts a Google Compute Engine (GCE) Persistent Disk into the pod. Content preserved on pod delete → prepopulate, data “hand off”
  - awsElasticBlockStore - mounts an Amazon Web Services EBS Volume into the pod. Content preserved.
  - nfs – allows an existing NFS share to be mounted into the pod. Allows multiple writers. The server should be configured. Content is preserved.
  - iscsi – single writer. Can be mounted read only by multiple pods.
  - glusterfs – multiple writers.
  - rbd - single writer. Can be mounted read only by multiple pods.
  - cephfs – multiple writers.
  - secret
  - persistentVolumeClaim

Persistent Volumes

- PersistentVolume (PV) – a cluster resource that hides the details of

storage implementation from the pod.
  - Can be of different types (HostPath, NFS, iSCSI, RBD, ... plugins)
  - Are independent from the pods that are using them.
- PersistentVolumeClaim (PVC) – a request for storage by a pod.
  - PVCs will consume PV resources.
  - PVC can request size, access mode, storage class.
- StorageClass – describes the "classes" of storages
  - Classes can map to quality-of-service levels, backup policies, ...
  - Allows for dynamic provisioning of Pvs.
- The pod definition will use the PVC for defining the volumes consumed by the containers.
- Dynamic provisioning is possible using the StorageClass definition.
  - A StorageClass will contain the *provisioner* and *parameter* fields.

https://kubernetes.io/docs/concepts/storage/volumes/

```
================================
PERSISTENT VOLUME
================================
apiVersion: v1
kind: PersistentVolume
metadata:
  name: myebsvol
spec:
  capacity:
    storage: 1Gi
  accessModes:
    - ReadWriteOnce
  persistentVolumeReclaimPolicy: Recycle
  awsElasticBlockStore:
    volumeID:          # YAHAN APNI EBS VOLUME ID DAALO
    fsType: ext4
============
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: myebsvolclaim
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
================================================================================
==================================
apiVersion: apps/v1
kind: Deployment
metadata:
  name: pvdeploy
spec:
  replicas: 1
  selector:      # tells the controller which pods to watch/belong to
    matchLabels:
      app: mypv
  template:
    metadata:
      labels:
        app: mypv
    spec:
      containers:
      - name: shell
        image: centos
        command: ["bin/bash", "-c", "sleep 10000"]
        volumeMounts:
        - name: mypd
```

```
      mountPath: "/tmp/persistent"
  volumes:
    - name: mypd
      persistentVolumeClaim:
        claimName: myebsvolclaim
```

https://kubernetes.io/docs/tasks/configure-pod-container/configure-liveness-readiness-startup-probes/

```
================================================================================
================================================
HEALTHCHECK/LIVENESSPROBE
================================================================================
================================================
apiVersion: v1
kind: Pod
metadata:
  labels:
    test: liveness
  name: mylivenessprobe
spec:
  containers:
  - name: liveness
    image: ubuntu
    args:
    - /bin/sh
    - -c
    - touch /tmp/healthy; sleep 1000
    livenessProbe:
      exec:
        command:
        - cat
        - /tmp/healthy
      initialDelaySeconds: 5
      periodSeconds: 5
      timeoutSeconds: 30
----------------------------------------
```